

# AURIX™ TC3xx

## About this document

### Scope and purpose

This User's Manual describes the Infineon AURIX™ TC3xx Platform family, a range of 32-bit multicore microcontrollers based on the Infineon TriCore™ Architecture.

This family document covers the superset functionality. It is supplemented by a separate device specific document "Appendix" that covers differences of a particular device to this family superset.

### Attention

For the devices TC39x-B, TC38x, TC37x, TC37xEXT, TC36x, TC35x, TC3Ex, TC33x/TC32x and TC33xEXT this set of documentation was released as User's Manual V1.5.0.

For the device TC3Ax this set of documentation qualifies only as Target Specification.

## Table of Contents

<b>About this document.....</b>	<b>Preface-1</b>
<b>Table of Contents .....</b>	<b>TOC-1</b>
<b>1 Introduction .....</b>	<b>1-1</b>
1.1 About this Document .....	1-1
1.1.1 Related Documentations .....	1-1
1.1.2 Text Conventions .....	1-1
1.1.3 Family Specification and Appendix .....	1-2
1.1.4 Register and Memory Address Documentation .....	1-2
1.1.5 Reserved, Undefined, and Unimplemented Terminology .....	1-3
1.1.6 Register Access Modes .....	1-3
1.1.7 Register Reset Documentation .....	1-5
1.1.8 Abbreviations and Acronyms .....	1-7
1.2 System Architecture of the AURIX™ TC3xx Platform .....	1-10
1.2.1 AURIX™ TC3xx Platform High End – TC39x .....	1-11
1.2.2 AURIX™ TC3xx Platform - Family Overview .....	1-12
1.2.2.1 TC38x Block Diagram .....	1-20
1.2.2.2 TC37xEXT Block Diagram .....	1-21
1.2.2.3 TC37x Block Diagram .....	1-22
1.2.2.4 TC36x Block Diagram .....	1-23
1.2.2.5 TC35x Block Diagram .....	1-24
1.2.2.6 TC33xEXT Block Diagram .....	1-25
1.2.2.7 TC33x Block Diagram .....	1-26
1.2.2.8 TC3Ex Block Diagram .....	1-27
1.2.2.9 TC3Ax Block Diagram .....	1-28
1.2.3 Variants .....	1-29
1.2.3.1 Encoding of the Product Name .....	1-29
1.2.3.2 Emulation Devices .....	1-31
1.2.3.3 TC32x .....	1-31
1.2.3.4 TC39x With Feature Package “P” .....	1-32
1.2.4 Revision History .....	1-32
1.3 On-Chip Debug Support (OCDS) .....	1-35
1.3.1 Introduction .....	1-35
1.3.2 Feature List .....	1-36
1.3.3 Family Overview .....	1-39
1.3.4 Tool Interface Recommendations .....	1-39
1.3.5 Debug Access Server (DAS) .....	1-40
1.3.6 Revision History .....	1-40
1.4 Emulation Device (ED) .....	1-41
1.4.1 Block Diagram .....	1-42
1.4.2 Feature List .....	1-44
1.4.3 Comparison to AURIX Emulation Devices .....	1-46
1.4.4 Trace Source Multiplexer .....	1-47
1.4.4.1 TMUX Setting Options .....	1-47
1.4.4.2 Trace Source Multiplexer in CPU Subsystem (TCMUX) .....	1-48
1.4.4.3 Parallel Trace Use Cases .....	1-49
1.4.5 DAP ED Interface (DAPE) .....	1-50
1.4.6 Revision History .....	1-51

1.5	Software over the Air (SOTA) .....	1-52
1.5.1	Overview .....	1-52
1.5.2	Functional Description .....	1-52
1.5.2.1	Performance considerations .....	1-52
1.5.2.2	Configuring for SOTA .....	1-52
1.5.2.2.1	Configuration parameters .....	1-53
1.5.2.2.2	Initial device configuration for SOTA .....	1-53
1.5.2.2.3	Runtime SWAP configuration .....	1-54
1.5.3	Safety .....	1-57
1.5.4	Security .....	1-57
1.5.5	Revision History .....	1-57
<b>2</b>	<b>Memory Maps (MEMMAP)</b> .....	<b>2-1</b>
2.1	Feature List .....	2-1
2.2	Overview .....	2-1
2.3	Functional Description .....	2-2
2.3.1	Segments .....	2-2
2.3.2	Address Map of the On Chip Bus System .....	2-3
2.3.2.1	Segments 0 to 14 .....	2-3
2.3.2.2	Segment 15 .....	2-14
2.3.3	Memory Accesses .....	2-22
2.4	Revision History .....	2-23
<b>3</b>	<b>AURIX™ TC3xx Platform Firmware</b> .....	<b>3-1</b>
3.1	Functional description .....	3-1
3.1.1	Startup Software .....	3-1
3.1.1.1	Events triggering SSW execution .....	3-1
3.1.1.1.1	Cold (initial) power-on reset .....	3-1
3.1.1.1.2	System reset .....	3-2
3.1.1.1.3	Application reset .....	3-2
3.1.1.2	Clock system during start-up .....	3-2
3.1.1.3	RAM overwrite during start-up .....	3-3
3.1.1.4	Stand-by controller handling during start-up .....	3-3
3.1.1.5	Boot Options Summary .....	3-3
3.1.1.6	Boot Mode evaluation sequence .....	3-4
3.1.1.6.1	Evaluation of Boot Mode Headers .....	3-4
3.1.1.6.2	Alternate Boot Mode evaluation .....	3-9
3.1.1.6.3	Processing in case no valid BMHD found .....	3-11
3.1.1.6.4	Processing in case no Boot Mode configured by SSW .....	3-12
3.1.1.7	Startup Software Main Flow .....	3-14
3.1.1.7.1	Flash ramp-up .....	3-15
3.1.1.7.2	Device Configuration .....	3-15
3.1.1.7.3	RAM Initialization .....	3-15
3.1.1.7.4	Select and execute Startup Modes .....	3-15
3.1.1.7.5	LBIST execution .....	3-15
3.1.1.7.6	Lockstep configuration .....	3-16
3.1.1.7.7	Debug System handling .....	3-16
3.1.1.7.8	ESR0 pin handling .....	3-17
3.1.1.7.9	Ending the SSW and Starting the User Code .....	3-17
3.1.2	Checker Software .....	3-19
3.1.2.1	CHSW execution flow .....	3-19

3.1.2.2	Checks performed by CHSW and exit information .....	3-20
3.1.2.3	Checker Software exit information for ALL CHECKS PASSED .....	3-22
3.1.3	Bootstrap Loaders .....	3-23
3.1.3.1	ASC Bootstrap loader .....	3-23
3.1.3.2	CAN Bootstrap loader .....	3-24
3.1.3.2.1	CAN BSL summary .....	3-24
3.1.3.2.2	Clock system during CAN BSL .....	3-24
3.1.3.2.3	CAN BSL usage after application reset .....	3-25
3.1.3.2.4	Supported CAN features .....	3-25
3.1.3.2.5	CAN BSL flow .....	3-26
3.1.4	Support for Software over the Air (SOTA) .....	3-30
3.1.5	Shutdown request handler .....	3-30
3.1.6	Power Supply Friendly Debug Monitor .....	3-31
3.2	Registers .....	3-32
3.2.1	Firmware specific usage of device registers .....	3-32
3.2.1.1	Registers providing information on the boot selections .....	3-32
3.2.1.2	Registers providing information on the Checker Software activity .....	3-34
3.3	Revision History .....	3-43
<b>4</b>	<b>On-Chip System Connectivity {and Bridges} .....</b>	<b>4-1</b>
4.1	Feature List .....	4-2
4.1.1	What is new in the SRI Fabric .....	4-2
4.2	Overview .....	4-2
4.3	Functional Description .....	4-5
4.3.1	Operational Overview .....	4-5
4.3.1.1	Master Connection Interface (MCI) .....	4-6
4.3.1.2	Slave Connection Interface (SCI) .....	4-6
4.3.1.3	Slave Arbiter .....	4-6
4.3.1.4	Default Slave .....	4-7
4.3.1.5	OnLine Data Acquisition (OLDA) .....	4-7
4.3.1.6	Control and Status Register Access Protection .....	4-8
4.3.2	Arbitration Details .....	4-8
4.3.2.1	Master Request Arbitration .....	4-8
4.3.3	SRI Errors .....	4-11
4.3.3.1	SRI Protocol Errors .....	4-11
4.3.3.2	SRI Transaction ID Errors .....	4-11
4.3.3.3	SRI EDC Errors .....	4-12
4.3.3.4	Error Handling .....	4-12
4.3.3.5	Error Tracking Capability .....	4-13
4.3.3.6	Indication Event Interactions .....	4-14
4.3.3.7	Releasing the lock from registers ERR and ERRADD .....	4-14
4.3.4	Implementation of the SRI Fabric .....	4-14
4.3.4.1	Mapping of SRI Masters to Domain 0 Master Interfaces .....	4-15
4.3.4.2	Mapping of SRI Slaves to Domain 0 Slave Interfaces .....	4-16
4.3.4.3	Mapping of SRI Masters to Domain 1 Master Interfaces .....	4-17
4.3.4.4	Mapping of SRI Slaves to Domain 1 Slave Interfaces .....	4-17
4.3.4.5	Mapping of SRI Masters to Domain 2 Master Interfaces .....	4-17
4.3.4.6	Mapping of SRI Slaves to Domain 2 Slave Interfaces .....	4-18
4.4	Registers .....	4-18
4.4.1	Domain Common Registers .....	4-20

4.4.2	SCI Control Registers .....	4-25
4.5	S2S Bridge .....	4-29
4.5.1	EDC Errors .....	4-29
4.5.2	Protocol Errors .....	4-29
4.5.3	Transaction ID Errors .....	4-29
4.5.4	Transaction Errors for Writes via S2S Bridge .....	4-29
4.6	SFI_F2S Bridge .....	4-30
4.6.1	Functional Overview .....	4-30
4.7	SFI_S2F Bridge .....	4-31
4.7.1	Functional Overview .....	4-31
4.8	Resource Access Times .....	4-31
4.9	Revision History .....	4-34
4.10	FPI Interconnect .....	4-35
4.10.1	Feature List .....	4-35
4.10.1.1	Delta to TC2xx .....	4-36
4.10.2	Overview .....	4-36
4.10.2.1	Bus Transaction Types .....	4-36
4.10.2.2	Reaction of a Busy Slave .....	4-37
4.10.2.3	Address Alignment Rules .....	4-37
4.10.2.4	FPI Bus Basic Operations .....	4-37
4.10.3	Functional Description (SBCU, EBCU) .....	4-38
4.10.3.1	FPI Bus Arbitration .....	4-38
4.10.3.1.1	Arbitration on the System Peripheral Bus .....	4-39
4.10.3.1.2	Default Master .....	4-39
4.10.3.1.3	Arbitration Algorithms .....	4-39
4.10.3.2	FPI Bus Error Handling .....	4-40
4.10.4	FPI Bus Integrity Support .....	4-41
4.10.4.1	Safety Support .....	4-43
4.10.4.2	FPI EDC Overview .....	4-45
4.10.4.3	Error Injection .....	4-46
4.10.4.4	SPB: Mapping of ALARM signals to SBCU_ALSTATx and SBCU_ALCLRx registers .....	4-46
4.10.4.5	BBB: Mapping of ALARM signals to EBCU_ALSTATx and EBCU_ALCLRx registers .....	4-49
4.10.5	Debug .....	4-51
4.10.5.1	Address Trigger .....	4-51
4.10.5.2	Signal Status Trigger .....	4-52
4.10.5.3	Grant Trigger .....	4-53
4.10.5.4	Combination of Trigger Events .....	4-54
4.10.5.5	BCU Breakpoint Generation Examples .....	4-54
4.10.6	Registers .....	4-56
4.10.6.1	Registers Description .....	4-57
4.10.6.2	System Registers .....	4-73
4.10.6.3	Register Access Protection (ACCEN1/0) .....	4-74
4.10.6.4	Kernel Reset Registers (KRST1/0, KRSTCLR) .....	4-75
4.10.6.5	Clock Control Register (CLC) .....	4-75
4.10.6.6	OCDS Control and Status Register (OCS) .....	4-75
4.10.7	On Chip Bus Master TAG Assignments .....	4-75
4.10.8	Revision History .....	4-77
<b>5</b>	<b>CPU Subsystem .....</b>	<b>5-1</b>
5.1	Feature List .....	5-2

5.2	Overview .....	5-4
5.2.1	CPU Diagram .....	5-5
5.2.2	Instruction Fetch Unit .....	5-5
5.2.3	Execution Unit .....	5-6
5.2.4	General Purpose Register File .....	5-7
5.3	Functional Description .....	5-7
5.3.1	Summary of functional changes from AURIX .....	5-7
5.3.2	Summary of changes from TC39x A-Step .....	5-8
5.3.3	AURIX™ Family CPU configurations .....	5-9
5.3.4	CPU Implementation-Specific Features .....	5-12
5.3.4.1	Context Save Areas / Context Operations .....	5-12
5.3.4.2	Program Counter (PC) Register .....	5-12
5.3.4.3	Store Buffers .....	5-12
5.3.4.4	Interrupt System .....	5-14
5.3.4.5	Trap System .....	5-14
5.3.4.6	WAIT Instruction .....	5-15
5.3.4.7	Invalid Opcode .....	5-15
5.3.4.8	Speculation extent .....	5-15
5.3.4.9	Instruction Memory Range Limitations .....	5-16
5.3.4.10	Atomicity of Data Accesses .....	5-16
5.3.4.11	A11 usage .....	5-17
5.3.4.12	Independent Core Kernel Reset .....	5-17
5.3.4.12.1	Kernel Reset Registers .....	5-18
5.3.4.13	CPU Clock Control .....	5-21
5.3.4.14	CPU Core Special Function Registers (CSFR) .....	5-22
5.3.4.14.1	Registers .....	5-22
5.3.4.15	CPU General Purpose Registers .....	5-31
5.3.4.15.1	CPU General Purpose Registers .....	5-31
5.3.4.16	FPU Registers .....	5-32
5.3.4.16.1	FPU registers .....	5-32
5.3.4.17	CPU Memory Protection Registers .....	5-36
5.3.4.18	Temporal Protection Registers .....	5-38
5.3.4.18.1	Temporal Protection Registers .....	5-38
5.3.4.19	Exception Timer .....	5-39
5.3.4.19.1	Exception Timers Registers .....	5-42
5.3.4.20	Memory Integrity Registers .....	5-46
5.3.4.20.1	Register Descriptions .....	5-46
5.3.4.21	CPU Core Debug and Performance Counter Registers .....	5-53
5.3.4.21.1	Counter Source Details .....	5-53
5.3.4.22	CPU Subsystem Register Summary .....	5-55
5.3.4.22.1	Summary of CSFR Reset Values and Access Modes .....	5-56
5.3.4.22.2	Summary of SFR Reset Values and Access modes .....	5-61
5.3.5	CPU Instruction Timing .....	5-64
5.3.5.1	Integer-Pipeline Instructions .....	5-65
5.3.5.1.1	Simple Arithmetic Instruction Timings .....	5-65
5.3.5.1.2	Multiply Instruction Timings .....	5-68
5.3.5.1.3	Multiply Accumulate (MAC) Instruction Timing .....	5-68
5.3.5.1.4	Control Flow Instruction Timing .....	5-69
5.3.5.2	Load-Store Pipeline Instructions .....	5-70
5.3.5.2.1	Address Arithmetic Timing .....	5-70

5.3.5.2.2	CSA Control Flow Instruction Timing .....	5-71
5.3.5.2.3	Load Instruction Timing .....	5-71
5.3.5.2.4	Store Instruction Timing .....	5-72
5.3.5.3	Floating Point Pipeline Timing .....	5-73
5.3.6	Local Memory Details .....	5-73
5.3.6.1	Memory Addressing .....	5-74
5.3.6.1.1	Local and Global Addressing .....	5-74
5.3.6.1.2	CSFR and SFR base Locations .....	5-74
5.3.6.1.3	Cache Memory Access .....	5-75
5.3.6.1.4	Customer-ID Numbering .....	5-75
5.3.6.2	Memory Integrity Error Handling .....	5-76
5.3.6.2.1	Program Side Memories .....	5-76
5.3.6.2.2	Data Side Memories .....	5-77
5.3.6.2.3	Memory Initialisation .....	5-79
5.3.6.3	Program Memory Interface (PMI) .....	5-80
5.3.6.3.1	TC1.6.2P PMI Description .....	5-80
5.3.6.3.2	PMI Registers .....	5-83
5.3.6.4	Data Memory Interface (DMI) .....	5-86
5.3.6.4.1	DMI Description .....	5-86
5.3.6.4.2	Distributed LMU (DLMU) .....	5-88
5.3.6.4.3	DMI Trap Generation .....	5-88
5.3.6.4.4	DMI Registers .....	5-89
5.3.7	Miscellaneous .....	5-93
5.3.7.1	Boot Halt .....	5-93
5.3.7.2	SSH usage recommendations .....	5-93
5.3.7.3	Debug restrictions .....	5-94
5.3.7.4	Local Pflash Bank Configuration Registers .....	5-94
5.3.7.4.1	Registers .....	5-95
5.3.8	Lockstep Comparator Logic (LCL) .....	5-100
5.3.8.1	Feature List .....	5-100
5.3.8.2	Lockstep Control .....	5-101
5.3.8.3	Lockstep Monitoring .....	5-101
5.3.8.4	Lockstep Self Test .....	5-102
5.3.8.5	Lockstep Failure Signalling Test .....	5-103
5.3.8.6	Functional Redundancy .....	5-103
5.3.9	Data Access Overlay (OVC) .....	5-104
5.3.9.1	Data Access Redirection .....	5-105
5.3.9.2	Target Memories .....	5-106
5.3.9.2.1	Online Data Acquisition (OLDA) Space .....	5-106
5.3.9.3	Overlay Memories .....	5-107
5.3.9.3.1	Local Memory .....	5-107
5.3.9.3.2	External Memory .....	5-107
5.3.9.3.3	DSPR & PSPR Memory .....	5-107
5.3.9.4	Global Overlay Control .....	5-107
5.3.9.4.1	Global Overlay Control Synchronisation .....	5-108
5.3.9.5	Overlay Configuration Change .....	5-108
5.3.9.6	Access Protection, Attributes, Concurrent Matches .....	5-109
5.3.9.7	Overlay Control Registers .....	5-110
5.3.9.7.1	Block control registers .....	5-110
5.3.9.8	Global overlay control registers .....	5-114

5.3.10	CPU Architecture registers .....	5-115
5.3.10.1	Registers with architecturally defined reset values .....	5-115
5.3.10.2	Program Counter (PC) .....	5-115
5.3.10.3	Registers with Implementation specific reset values .....	5-116
5.4	Safety Measures .....	5-124
5.4.1	SRI Bus Master Address Phase Error Injection .....	5-124
5.4.2	SRI Bus Master Write Phase Error Injection .....	5-124
5.4.3	SRI bus Slave Read Phase Error Injection .....	5-124
5.4.4	SRI Error Capture .....	5-124
5.4.5	SRI Safe Data Master tag .....	5-125
5.4.6	Safety Protection System .....	5-125
5.4.6.1	Bus MPU .....	5-125
5.4.6.2	Register Access Enable Protection .....	5-126
5.4.7	Registers Implementing Safety Features .....	5-127
5.4.7.1	SRI safety registers .....	5-128
5.4.7.2	Safety Protection registers .....	5-129
5.5	IO Interfaces .....	5-138
5.6	Revision History .....	5-138
<b>6</b>	<b>Non Volatile Memory (NVM) Subsystem .....</b>	<b>6-1</b>
6.1	Overview .....	6-1
6.2	Functional Description .....	6-3
6.2.1	Definition of Terms .....	6-3
6.2.2	Major changes from Aurix to AURIXTC3XX .....	6-4
6.2.3	Flash Structure .....	6-4
6.2.3.1	Program Flash Banks .....	6-5
6.2.3.1.1	PFLASH Tuning Protection (TP) and HSM Support .....	6-5
6.2.3.1.2	Erase Counters .....	6-7
6.2.3.2	EEPROM Emulation with DFLASH .....	6-8
6.2.3.2.1	DFLASH Emulation Modes .....	6-8
6.2.3.2.2	Robust EEPROM Emulation .....	6-8
6.2.3.3	Data Flash Bank DFLASH0 .....	6-9
6.2.3.4	Data Flash Bank DFLASH1 .....	6-12
6.2.4	Program Flash (PFLASH) Features .....	6-13
6.2.5	Data Flash (DFLASH) Features .....	6-14
6.2.6	Boot ROM (BROM) Features .....	6-15
6.3	Safety Measures .....	6-16
6.3.1	Safety Endinit protection .....	6-16
6.3.2	Access Control .....	6-16
6.3.3	Data Reliability and Integrity .....	6-17
6.3.3.1	PFLASH ECC .....	6-17
6.3.3.2	DFLASH ECC .....	6-18
6.3.4	Integrity of Pflash read data wait cycles .....	6-18
6.3.5	Alarms .....	6-18
6.3.5.1	SRI Access Address Phase Error .....	6-19
6.3.5.2	SRI Access Write Data Phase Error .....	6-19
6.4	Revision History .....	6-20
6.5	Data Memory Unit (DMU) .....	6-21
6.5.1	Overview .....	6-21
6.5.2	Functional Description .....	6-22

6.5.2.1	Flash Read Access .....	6-22
6.5.2.1.1	Transaction Types .....	6-22
6.5.2.1.2	Configuring Flash Read Access Cycles .....	6-22
6.5.2.2	Flash Operations .....	6-23
6.5.2.2.1	Page Mode .....	6-24
6.5.2.2.2	Command Sequences .....	6-24
6.5.2.2.3	Command Sequence Definitions .....	6-27
6.5.2.2.4	Protection for Verify Command Sequences .....	6-38
6.5.2.2.5	DMU Commands .....	6-39
6.5.2.2.6	Suspend and Resume Operations .....	6-39
6.5.2.2.7	Programming Voltage Selection .....	6-43
6.5.2.2.8	Performing Flash Operations .....	6-43
6.5.2.3	Traps .....	6-47
6.5.2.4	Interrupts .....	6-47
6.5.2.4.1	Host Command Interface .....	6-47
6.5.2.4.2	HSM Command Interface .....	6-47
6.5.2.5	Error Handling .....	6-48
6.5.2.5.1	Handling Errors During Startup .....	6-48
6.5.2.5.2	Handling Errors During Operation .....	6-48
6.5.2.6	DMU Modes .....	6-51
6.5.2.6.1	Operation Mode .....	6-52
6.5.2.6.2	Error Mode .....	6-52
6.5.2.6.3	Power modes .....	6-52
6.5.2.7	Internal Connections .....	6-53
6.5.2.7.1	Clocks .....	6-53
6.5.2.7.2	Interrupts and Service Requests .....	6-53
6.5.2.7.3	Cross Triggers .....	6-54
6.5.2.8	Power Modes .....	6-55
6.5.2.8.1	Flash Prefetch Buffers .....	6-55
6.5.2.8.2	Demand Mode .....	6-55
6.5.2.8.3	Dynamic Idle Mode .....	6-55
6.5.2.8.4	Sleep Mode .....	6-55
6.5.2.8.5	Cranking Mode .....	6-55
6.5.2.8.6	Standby Mode .....	6-56
6.5.2.9	Boot ROM (BROM) .....	6-56
6.5.2.9.1	Read Accesses .....	6-56
6.5.2.9.2	Data Integrity .....	6-56
6.5.3	Registers .....	6-57
6.5.3.1	Flash ID and BootROM Registers (PMU) .....	6-58
6.5.3.1.1	PMU Identification .....	6-58
6.5.3.2	DMU Registers .....	6-59
6.5.3.2.1	DMU Identification .....	6-63
6.5.3.2.2	Host Command Interface .....	6-63
6.5.3.2.3	Flash Error Registers .....	6-78
6.5.3.2.4	Data Flash Bank 0 ECC Registers .....	6-82
6.5.3.2.5	Data Flash Bank 0 Mode Control Registers .....	6-87
6.5.3.2.6	Power Mode Registers .....	6-88
6.5.3.2.7	PFLASH Protection Configuration .....	6-92
6.5.3.2.8	Tuning Protection Configuration .....	6-93
6.5.3.2.9	DFLASH Protection Configuration .....	6-94

6.5.3.2.10	Suspend .....	6-99
6.5.3.2.11	Margin Check Control .....	6-100
6.5.3.2.12	Access Protection Registers .....	6-101
6.5.3.2.13	Protection Configuration .....	6-102
6.5.3.2.14	HSM Command Interface .....	6-118
6.5.3.2.15	HSM Flash Error Registers .....	6-121
6.5.3.2.16	Data Flash Bank 1 ECC Registers .....	6-123
6.5.3.2.17	Data Flash Bank 1 Mode Control Registers .....	6-129
6.5.3.2.18	HSM Suspend .....	6-129
6.5.3.2.19	Margin Check Control .....	6-130
6.5.3.2.20	HSM OTP Protection Configuration .....	6-131
6.5.3.2.21	HSM Interface Protection Configuration .....	6-138
6.5.4	Security .....	6-140
6.5.4.1	Effective Flash Read Protection .....	6-140
6.5.4.1.1	PFLASH Read Protection .....	6-141
6.5.4.1.2	DFLASH Read Protection .....	6-141
6.5.4.2	Effective Flash Write Protection .....	6-142
6.5.4.2.1	PFLASH Write Protection .....	6-142
6.5.4.2.2	DFLASH Write Protection .....	6-142
6.5.4.3	Configuring Protection in the UCB .....	6-143
6.5.4.3.1	UCB Confirmation .....	6-143
6.5.4.3.2	UCB_BMHDX_ORIG and UCB_BMHDX_COPY ( $x = 0-3$ ) .....	6-145
6.5.4.3.3	UCB_SSW .....	6-146
6.5.4.3.4	UCB_USER .....	6-147
6.5.4.3.5	UCB_TEST .....	6-148
6.5.4.3.6	UCB_HSMCFG .....	6-148
6.5.4.3.7	UCB_REDSEC .....	6-149
6.5.4.3.8	UCB_PFLASH_ORIG and UCB_PFLASH_COPY .....	6-149
6.5.4.3.9	UCB_DFLASH_ORIG and UCB_DFLASH_COPY .....	6-150
6.5.4.3.10	UCB_DBG_ORIG and UCB_DBG_COPY .....	6-151
6.5.4.3.11	UCB_HSM_ORIG and UCB_HSM_COPY .....	6-152
6.5.4.3.12	UCB_HSMCOTP0/1_ORIG and UCB_HSMCOTP0/1_COPY .....	6-153
6.5.4.3.13	UCB_ECPPIO_ORIG and UCB_ECPPIO_COPY .....	6-155
6.5.4.3.14	UCB_SWAP_ORIG and UCB_SWAP_COPY .....	6-156
6.5.4.3.15	UCB_OTPy_ORIG and UCB_OTPy_COPY ( $y = 0-7$ ) .....	6-157
6.5.4.3.16	Spare UCB .....	6-159
6.5.4.4	System Wide Effects of Flash Protection .....	6-159
6.5.4.4.1	HSM Booting .....	6-159
6.5.4.4.2	Destructive Debug Entry .....	6-160
6.5.5	Revision History .....	6-161
6.6	Program Flash Interface (PFI) .....	6-162
6.6.1	Overview .....	6-162
6.6.2	Functional Description .....	6-163
6.6.2.1	Demand Path .....	6-163
6.6.2.2	Data Read Line Buffer (DRLB) .....	6-163
6.6.2.3	Flash Prefetch Buffer (FPB) .....	6-163
6.6.3	Erase Counter and Register Accesses .....	6-163
6.6.3.1	Erase Counter .....	6-164
6.6.3.2	User Registers .....	6-164
6.6.4	Safety Measures .....	6-165

6.6.4.1	Access Enable .....	6-165
6.6.4.2	ECC encoding of read data to CPU .....	6-165
6.6.4.3	ECC error detection of wait cycle configuration from DMU .....	6-165
6.6.4.4	PFI Partial Lockstep (PPL) .....	6-165
6.6.4.5	Busy checker .....	6-165
6.6.5	Revision History .....	6-166
6.7	Non Volatile Memory (NVM) .....	6-167
6.7.1	Overview .....	6-167
6.7.2	Functional Description of the Flash Standard Interface (FSI) .....	6-169
6.7.2.1	FSI ROM .....	6-169
6.7.2.2	FSI SFR .....	6-169
6.7.2.2.1	FSI SFR Access Control .....	6-169
6.7.2.3	Communication with FSI .....	6-170
6.7.2.3.1	DMU Command Sequences .....	6-170
6.7.3	Registers .....	6-170
6.7.3.1	FSI Registers .....	6-171
6.7.3.1.1	Status register .....	6-171
6.7.3.2	PFRWB (PFI) Registers .....	6-173
6.7.3.2.1	PFI ECC Registers .....	6-174
6.7.3.2.2	PFI Corrected Single Bits Address Buffer (SBAB) .....	6-177
6.7.3.2.3	PFI Corrected Double Bits Address Buffer (DBAB) .....	6-178
6.7.3.2.4	PFI Uncorrected Multi Bits Address Buffer (MBAB) .....	6-179
6.7.3.2.5	PFI Uncorrected All Zeros Bits Address Buffer (ZBAB) .....	6-180
6.7.4	Revision History .....	6-181
6.8	User Configuration Block (UCB) .....	6-182
6.8.1	Overview .....	6-182
6.8.2	UCB Address Map .....	6-182
6.8.2.1	List of Defined UCBs .....	6-182
6.8.2.2	UCB_BMHDx_ORIG and UCB_BMHDx_COPY (x = 0 - 3) .....	6-183
6.8.2.3	UCB_SSW .....	6-185
6.8.2.4	UCB_USER .....	6-185
6.8.2.5	UCB_RETEST .....	6-187
6.8.2.6	UCB_PFLASH_ORIG and UCB_PFLASH_COPY .....	6-187
6.8.2.7	UCB_DFLASH_ORIG and UCB_DFLASH_COPY .....	6-188
6.8.2.8	UCB_DBG_ORIG and UCB_DBG_COPY .....	6-189
6.8.2.9	UCB_HSM_ORIG and UCB_HSM_COPY .....	6-190
6.8.2.10	UCB_HSMCOTP0/1_ORIG and UCB_HSMCOTP0/1_COPY .....	6-190
6.8.2.11	UCB_ECPPIO_ORIG and UCB_ECPPIO_COPY .....	6-191
6.8.2.12	UCB_SWAP_ORIG and UCB_SWAP_COPY .....	6-191
6.8.2.13	UCB OTPy_ORIG and UCB OTPy_COPY (y = 0 - 7) .....	6-192
6.8.2.14	UCB_REDSEC .....	6-193
6.8.3	UCB Entries .....	6-195
6.8.3.1	UCB_USER .....	6-195
6.8.3.2	UCB_SWAP_ORIG and UCB_SWAP_COPY .....	6-196
6.8.3.3	UCB_REDSEC .....	6-198
6.8.4	Revision History .....	6-203
<b>7</b>	<b>Local Memory Unit (LMU)</b> .....	<b>7-1</b>
7.1	Feature List .....	7-1
7.2	Functional Description .....	7-1

7.2.1	Local Memory (LMU SRAM) .....	7-1
7.2.2	Memory Protection .....	7-2
7.2.3	LMU Register Protection .....	7-2
7.2.4	Error Detection and Signalling .....	7-3
7.2.4.1	SRI access address phase error .....	7-3
7.2.4.2	SRI write access data phase error .....	7-3
7.2.4.3	Uncorrected ECC Error .....	7-3
7.2.4.4	SRAM Data Correction ECC failure .....	7-3
7.2.4.5	Internal Data Transfer ECC Error .....	7-3
7.2.4.6	Access Protection Violation .....	7-3
7.2.4.7	Internal SRAM Read Error .....	7-3
7.2.4.8	Control Logic Failure .....	7-3
7.2.5	SRAM Data Correction ECC failure .....	7-4
7.2.6	Internal Data Transfer ECC Error .....	7-4
7.2.7	Internal SRAM Read Error .....	7-4
7.2.8	Clock Control .....	7-4
7.3	LMURegisters .....	7-5
7.4	IO Interfaces .....	7-14
7.5	Revision History .....	7-15
<b>8</b>	<b>Default Application Memory (DAM)</b> .....	<b>8-1</b>
8.1	Feature List .....	8-1
8.2	Functional Description .....	8-1
8.2.1	Local Memory (DAM SRAM) .....	8-1
8.2.2	Memory Protection .....	8-2
8.2.3	DAM Register Protection .....	8-2
8.2.4	Error Detection and Signalling .....	8-3
8.2.4.1	SRI access address phase error .....	8-3
8.2.4.2	SRI write access data phase error .....	8-3
8.2.4.3	Uncorrected ECC Error .....	8-3
8.2.4.4	Access Protection Violation .....	8-3
8.2.5	Clock Control .....	8-3
8.3	Registers .....	8-4
8.4	Revision History .....	8-12
<b>9</b>	<b>System Control Units (SCU)</b> .....	<b>9-1</b>
9.1	Reset Control Unit (RCU) .....	9-3
9.1.1	Feature List .....	9-3
9.1.1.1	Delta to AURIX .....	9-3
9.1.2	Overview .....	9-3
9.1.2.1	Reset Triggers .....	9-3
9.1.2.2	Reset Types .....	9-4
9.1.2.3	Reset Sources Overview .....	9-4
9.1.2.4	Warm and Cold Resets .....	9-5
9.1.2.5	EVR Resets and PORST .....	9-5
9.1.2.6	Module Reset Behavior .....	9-5
9.1.3	Reset Controller Functional Description .....	9-6
9.1.3.1	Reset Generation .....	9-7
9.1.3.2	Shutdown and Reset Delay Timeout Counter (TOUTCNT) .....	9-7
9.1.3.3	Reset Triggers .....	9-8
9.1.3.3.1	Specific Reset Triggers .....	9-8

9.1.3.3.2	Configurable Reset Triggers .....	9-8
9.1.3.3.3	Prevention of Double SMU Resets .....	9-8
9.1.3.4	Debug Reset Specific Behavior .....	9-9
9.1.3.5	Module Resets .....	9-9
9.1.3.5.1	CPU Module Resets .....	9-10
9.1.3.6	Reset Controller Registers .....	9-10
9.1.3.6.1	Status Registers .....	9-10
9.1.3.6.2	Reset Configuration Registers .....	9-14
9.1.4	External Reset Sources and Indications .....	9-20
9.1.4.1	External Service Requests (ESRx) .....	9-20
9.1.4.1.1	ESRx as Reset Request Trigger .....	9-21
9.1.4.1.2	ESRx as Reset Output .....	9-21
9.1.4.1.3	ESR Registers .....	9-22
9.1.5	Boot Software Interface .....	9-31
9.1.5.1	Configuration done with Start-up .....	9-31
9.1.5.2	Start-up Configuration Options .....	9-31
9.1.5.3	Boot Software Registers .....	9-32
9.1.5.3.1	Start-up Status Registers .....	9-32
9.2	Trap Generation (TR) .....	9-35
9.2.1	Feature List .....	9-35
9.2.1.1	Delta to AURIX .....	9-35
9.2.2	Trap Handling .....	9-35
9.2.3	Trap Registers .....	9-37
9.3	System Register Unit (SRU) .....	9-43
9.3.1	Feature List .....	9-43
9.3.1.1	Delta to AURIX .....	9-43
9.3.2	Lockstep Comparator Logic Configuration .....	9-43
9.3.2.1	Lockstep Comparator Logic Control Registers .....	9-43
9.3.3	LBIST Support .....	9-48
9.3.3.1	Introduction .....	9-48
9.3.3.1.1	Functional Description .....	9-48
9.3.3.2	LBIST Control Register .....	9-51
9.3.4	Clock System Control registers .....	9-55
9.3.5	Global Overlay Controls .....	9-56
9.3.5.1	Global Overlay Control .....	9-56
9.3.6	Miscellaneous System Control .....	9-60
9.3.6.1	System Control Register .....	9-60
9.3.6.2	Identification Registers .....	9-61
9.3.6.3	Start-up Software Memory Registers .....	9-65
9.3.6.4	SCU Access Restriction Registers .....	9-68
9.3.6.5	SOTA Address Map Control .....	9-70
9.4	Watchdog Timers (WDT) .....	9-70
9.4.1	Feature List .....	9-70
9.4.1.1	Changes to AURIX TM Family .....	9-71
9.4.1.2	Changes from TC39x A-Step to AURIX TC3xx .....	9-71
9.4.2	Watchdog Timers Overview .....	9-71
9.4.2.1	Safety Watchdog .....	9-73
9.4.2.2	CPU Watchdogs .....	9-73
9.4.3	Features of the Watchdog Timers .....	9-74
9.4.4	The Endinit Functions .....	9-74

9.4.4.1	Password Access to WDTxCON0 .....	9-75
9.4.4.1.1	Static Password .....	9-76
9.4.4.1.2	Automatic Password Sequencing .....	9-76
9.4.4.1.3	Time-Independent Password .....	9-77
9.4.4.1.4	Time Check Password .....	9-77
9.4.4.2	Check Access to WDTxCON0 .....	9-77
9.4.4.3	Modify Access to WDTxCON0 .....	9-78
9.4.4.4	Access to Endinit-Protected Registers .....	9-78
9.4.4.4.1	Access to Endinit-Protected Registers using WDT .....	9-78
9.4.4.4.2	Access to Endinit-Protected Registers without using WDT .....	9-78
9.4.5	Timer Operation .....	9-79
9.4.5.1	Timer Modes .....	9-79
9.4.5.2	WDT Alarm Request .....	9-80
9.4.5.3	WDT Operation During Power-Saving Modes .....	9-80
9.4.5.4	Suspend Mode Support .....	9-81
9.4.6	Watchdog Timer Registers .....	9-83
9.5	External Request Unit (ERU) .....	9-102
9.5.1	Feature List .....	9-102
9.5.1.1	Delta to AURIX .....	9-103
9.5.2	Introduction .....	9-103
9.5.3	REQxy Digital PORT Input Glitch Filter (FILT) .....	9-105
9.5.4	External Request Selector Unit (ERS) .....	9-105
9.5.5	Event Trigger Logic (ETL) .....	9-106
9.5.6	Connecting Matrix .....	9-108
9.5.7	Output Gating Unit (OGU) .....	9-109
9.5.7.1	Trigger Combination .....	9-110
9.5.7.2	Pattern Detection .....	9-110
9.5.7.3	Triggering SMU alarms .....	9-111
9.5.8	External Request Unit Registers .....	9-113
9.6	Emergency Stop (ES) .....	9-121
9.6.1	Feature List .....	9-121
9.6.2	Delta to AURIX .....	9-121
9.6.3	Port Triggered Emergency Stop .....	9-121
9.6.4	SMU Event Triggered Emergency Stop .....	9-122
9.6.5	Emergency Stop Register .....	9-123
9.7	Power Management Control Registers (PMC) .....	9-125
9.8	Registers .....	9-126
9.8.1	Safety Flip-Flops .....	9-129
9.9	IO Interfaces .....	9-129
9.10	Revision History .....	9-130
9.10.1	SCU Complete Revision History .....	9-130
<b>10</b>	<b>Clocking System .....</b>	<b>10-1</b>
10.1	Overview .....	10-1
10.2	Clocking System Registers Overview .....	10-2
10.2.1	Safety Flip-Flops .....	10-3
10.3	Clock Sources .....	10-3
10.3.1	Oscillator Circuit (OSC) .....	10-3
10.3.1.1	External Input Clock Mode .....	10-4
10.3.1.2	External Crystal / Ceramic Resonator Mode .....	10-4

10.3.1.3	Oscillator Circuit Control Register .....	10-6
10.3.1.4	Configuration of the Oscillator .....	10-9
10.3.1.5	Oscillator Watchdog .....	10-9
10.3.2	Back-up Clock .....	10-10
10.4	Clock Speed Up-Scaling (PLLs) .....	10-10
10.4.1	System Phase-Locked Loop (System PLL) Module .....	10-10
10.4.1.1	Features .....	10-10
10.4.1.2	System PLL Functional Description .....	10-12
10.4.1.3	System PLL Registers .....	10-14
10.4.2	Peripheral Phase-Locked Loop (Peripheral PLL) Module .....	10-17
10.4.2.1	Features .....	10-17
10.4.2.2	Peripheral PLL Functional Description .....	10-18
10.4.2.3	Peripheral PLL Registers .....	10-20
10.5	Clock Distribution (CCU) .....	10-23
10.5.1	Clock Control Unit .....	10-23
10.5.1.1	Basic Clock System Mechanisms .....	10-24
10.5.1.2	Clock Divider Limitations .....	10-27
10.5.1.3	CCU Registers .....	10-28
10.6	Clock Emergency Behavior .....	10-44
10.7	External Clock Output .....	10-44
10.7.1	Programmable Frequency Output for EXTCLK0 .....	10-44
10.7.1.1	Fractional Divider Operating Modes .....	10-45
10.7.2	Programmable Frequency Output for EXTCLK1 .....	10-46
10.7.3	Clock Output Control Register .....	10-47
10.8	Clock Generation Unit .....	10-50
10.9	Safety Measures .....	10-50
10.9.1	Clock Monitoring .....	10-50
10.9.1.1	Clock Monitor Registers .....	10-53
10.10	Use Cases .....	10-56
10.11	Revision History .....	10-61
<b>11</b>	<b>Power Management System (PMS)</b> .....	<b>11-1</b>
11.1	Overview .....	11-2
11.2	Functional Description .....	11-2
11.2.1	Power Supply Infrastructure and Supply Start-up .....	11-2
11.2.1.1	Supply Mode Selection .....	11-2
11.2.1.2	Supply Ramp-up and Ramp-down Behavior .....	11-8
11.2.1.2.1	Single Supply mode (a) .....	11-8
11.2.1.2.2	Single Supply mode (e) .....	11-10
11.2.1.2.3	External Supply mode (d) .....	11-12
11.2.1.2.4	External Supply mode (h) .....	11-14
11.2.1.2.5	HWCFG, P32.1 / VGATE1P, P32.0 / VGATE1N behavior during Start-up .....	11-16
11.2.1.3	PMS Infrastructure Components .....	11-18
11.2.1.3.1	Independent VEVRSB & VDDPD Supply domain and EVR Pre-Regulator (EVRPR) .....	11-18
11.2.1.3.2	Reference Voltage Generation : Secondary Bandgap Reference (SHPBG) .....	11-18
11.2.1.3.3	100 MHz Back-up Clock Source (fBACK) .....	11-18
11.2.1.4	Die Temperature Measurement .....	11-20
11.2.2	Power Supply Generation and Monitoring .....	11-21
11.2.2.1	Linear Regulator Mode (EVR33) .....	11-21
11.2.2.2	Step-down Regulator (EVRC) .....	11-22

11.2.2.2.1	EVRC Frequency and Phase Synchronization to CCU6/GTM Input .....	11-26
11.2.2.3	Components and Layout .....	11-28
11.2.2.4	External Supply Modes .....	11-35
11.2.2.5	Supply Voltage Monitoring .....	11-37
11.2.2.5.1	Primary under-voltage monitors and Cold PORST .....	11-38
11.2.2.5.2	Secondary over- and under-voltage monitors and alarm generation .....	11-40
11.2.2.5.3	Power Built In Self Test at Start-up (PBIST) .....	11-46
11.2.2.5.4	Secondary Monitor and Standby SMU Built in Self Test (MONBIST) .....	11-46
11.2.2.6	Interrupts .....	11-47
11.2.2.7	OCDS Trigger Bus (OTGB) Interface .....	11-48
11.2.2.7.1	ADC Monitor and Voltage Trigger Sets .....	11-48
11.2.2.7.2	EVR Control output Trigger Sets .....	11-49
11.2.3	Power Management .....	11-50
11.2.3.1	Power Management Overview .....	11-50
11.2.3.2	Idle Mode .....	11-54
11.2.3.2.1	Entering Idle Mode : .....	11-54
11.2.3.2.2	State during Idle mode .....	11-54
11.2.3.2.3	Exiting Idle mode .....	11-55
11.2.3.3	Sleep Mode .....	11-55
11.2.3.3.1	Entering Sleep Mode .....	11-55
11.2.3.3.2	State during Sleep Mode .....	11-56
11.2.3.3.3	Exiting Sleep Mode .....	11-58
11.2.3.4	Standby Mode .....	11-60
11.2.3.4.1	Standby Mode with only VEVRSB domain supplied and VEXT domain switched off .....	11-60
11.2.3.4.2	Standby Mode with both VEXT and VEVRSB supplied via common supply rail. ....	11-61
11.2.3.4.3	Standby RAM .....	11-62
11.2.3.4.4	VEXT Supply Monitor .....	11-62
11.2.3.4.5	Pin Wake-up Unit .....	11-62
11.2.3.4.6	Standby ContolleR (SCR) Interface .....	11-63
11.2.3.4.7	Wake-up Timer (WUT) .....	11-64
11.2.3.4.8	Entering Standby Mode (only VEVRSB domain supplied) .....	11-67
11.2.3.4.9	Entering Standby Mode (both VEVRSB and VEXT domain supplied) .....	11-70
11.2.3.4.10	State during Standby Mode .....	11-71
11.2.3.4.11	Exiting Standby Mode - Wake-up event .....	11-71
11.2.3.4.12	Exiting Standby Mode - Power Fail or Reset event .....	11-74
11.2.3.5	Load Jump Sequencing and Voltage Droop .....	11-75
11.2.3.6	Core Die Temperature Sensor (DTSC) .....	11-78
11.3	Registers .....	11-78
11.3.1	Power Management Control Registers (PMS) .....	11-79
11.3.1.1	Safety Flip-Flops .....	11-82
11.3.1.2	Power Supply Generation and Monitoring Control Registers .....	11-83
11.3.1.3	Die Temperature Sensor Registers .....	11-151
11.3.1.4	Standby and Wake-up Control Registers .....	11-154
11.3.1.5	OCDS Trigger Bus Configuration Registers (OTGB) .....	11-174
11.3.1.6	SMU_STDBY Registers .....	11-179
11.3.2	Power Management Control Registers (SCU) .....	11-180
11.3.2.1	Power Management Control and Status Registers .....	11-180
11.4	IO Interfaces .....	11-198
11.5	Revision History .....	11-200
11.5.1	Changes from AURIX 2G PMS V2.2.19 onwards .....	11-200

<b>12</b>	<b>Power Management System for Low-End (PMSLE) .....</b>	<b>12-1</b>
12.1	Overview .....	12-2
12.2	Functional Description .....	12-2
12.2.1	Power Supply Infrastructure and Supply Start-up .....	12-2
12.2.1.1	Supply Mode Selection .....	12-2
12.2.1.2	Supply Ramp-up and Ramp-down Behavior .....	12-9
12.2.1.2.1	Single Supply mode (a) .....	12-9
12.2.1.2.2	Single Supply mode (e) .....	12-11
12.2.1.2.3	External Supply mode (d) .....	12-13
12.2.1.2.4	External Supply mode (h) .....	12-15
12.2.1.2.5	EVRC, VCAPx behavior during Start-up .....	12-17
12.2.1.3	PMS Infrastructure Components .....	12-18
12.2.1.3.1	Independent VEVRSB & VDDPD Supply domain and EVR Pre-Regulator (EVRPR) .....	12-18
12.2.1.3.2	Reference Voltage Generation : Secondary Bandgap Reference (SHPBG) .....	12-18
12.2.1.3.3	100 MHz Back-up Clock Source (fBACK) .....	12-18
12.2.1.4	Die Temperature Measurement .....	12-20
12.2.2	Power Supply Generation and Monitoring .....	12-21
12.2.2.1	Linear Regulator Mode (EVR33) .....	12-21
12.2.2.2	Switch Capacitor Regulator (EVRC) .....	12-22
12.2.2.2.1	EVRC Supply Pins .....	12-26
12.2.2.2.2	VDD Connectivity .....	12-28
12.2.2.2.3	EVRC Frequency and Phase Synchronization to CCU6/GTM Input .....	12-29
12.2.2.3	Components and Layout .....	12-32
12.2.2.4	External Supply Modes .....	12-33
12.2.2.5	Supply Voltage Monitoring .....	12-35
12.2.2.5.1	Primary under-voltage monitors and Cold PORST .....	12-36
12.2.2.5.2	Secondary over- and under-voltage monitors and alarm generation .....	12-38
12.2.2.5.3	Power Built In Self Test at Start-up (PBIST) .....	12-44
12.2.2.5.4	Secondary Monitor and Standby SMU Built in Self Test (MONBIST) .....	12-44
12.2.2.6	Interrupts .....	12-46
12.2.2.7	OCDS Trigger Bus (OTGB) Interface .....	12-47
12.2.2.7.1	ADC Monitor and Voltage Trigger Sets .....	12-47
12.2.2.7.2	EVR Control output Trigger Sets .....	12-48
12.2.3	Power Management .....	12-49
12.2.3.1	Power Management Overview .....	12-49
12.2.3.2	Idle Mode .....	12-53
12.2.3.2.1	Entering Idle Mode: .....	12-53
12.2.3.2.2	State during Idle mode .....	12-53
12.2.3.2.3	Exiting Idle mode .....	12-54
12.2.3.3	Sleep Mode .....	12-54
12.2.3.3.1	Entering Sleep Mode .....	12-54
12.2.3.3.2	State during Sleep Mode .....	12-55
12.2.3.3.3	Exiting Sleep Mode .....	12-57
12.2.3.4	Standby Mode .....	12-59
12.2.3.4.1	Standby Mode with only VEVRSB domain supplied and VEXT domain switched off .....	12-59
12.2.3.4.2	Standby Mode with both VEXT and VEVRSB supplied via common supply rail. ....	12-60
12.2.3.4.3	Standby RAM .....	12-61
12.2.3.4.4	VEXT Supply Monitor .....	12-61
12.2.3.4.5	Pin Wake-up Unit .....	12-61
12.2.3.4.6	Standby ContolleR (SCR) Interface .....	12-62

12.2.3.4.7	Wake-up Timer (WUT) .....	12-63
12.2.3.4.8	Entering Standby Mode (only VEVRSB domain supplied) .....	12-66
12.2.3.4.9	Entering Standby Mode (both VEVRSB and VEXT domain supplied) .....	12-69
12.2.3.4.10	State during Standby Mode .....	12-70
12.2.3.4.11	Exiting Standby Mode - Wake-up event .....	12-70
12.2.3.4.12	Exiting Standby Mode - Power Fail or Reset event .....	12-73
12.2.3.5	Load Jump Sequencing and Voltage Droop .....	12-74
12.2.3.6	Core Die Temperature Sensor (DTSC) .....	12-77
12.3	Registers .....	12-78
12.3.1	Register Access Modes .....	12-78
12.3.2	Power Management Control Registers (PMS) .....	12-80
12.3.2.1	Safety Flip-Flops .....	12-82
12.3.2.2	Power Supply Generation and Monitoring Control Registers .....	12-83
12.3.2.3	Die Temperature Sensor Registers .....	12-146
12.3.2.4	Standby and Wake-up Control Registers .....	12-149
12.3.2.5	OCDS Trigger Bus Configuration Registers (OTGB) .....	12-169
12.3.2.6	SMU Registers .....	12-174
12.3.3	Power Management Control Registers (SCU) .....	12-175
12.3.3.1	Power Management Control and Status Registers .....	12-175
12.4	IO Interfaces .....	12-193
12.5	Revision History .....	12-195
12.5.1	Changes from AURIX TC33x PMS V1.0.1 Onwards .....	12-195
<b>13</b>	<b>Memory Test Unit (MTU)</b> .....	<b>13-1</b>
13.1	Feature List .....	13-1
13.2	Overview .....	13-1
13.3	Functional Description .....	13-2
13.3.1	Major Functional Changes from TC39xA-Step to TC39XB-Step / TC38XA-Step .....	13-2
13.3.2	SRAM Support Hardware (SSH) .....	13-3
13.3.3	Control and Status Interfaces .....	13-3
13.3.3.1	Interface to the CPU .....	13-3
13.3.4	Enabling the SRAM Support Hardware (SSH) .....	13-4
13.3.4.1	Security-Sensitive Memories and AutoInitialization .....	13-4
13.3.4.1.1	Security Applications .....	13-5
13.3.4.1.2	Non-Security Applications .....	13-5
13.3.4.2	Memory Map selection .....	13-5
13.3.5	SRAM Support Hardware (SSH) Operation .....	13-5
13.3.5.1	Memory Testing and Initialization .....	13-6
13.3.5.1.1	Starting a Memory Test Sequence .....	13-6
13.3.5.1.2	Memory Test Done Interrupt .....	13-6
13.3.5.1.3	Getting Detailed Memory Test Results .....	13-6
13.3.5.1.4	Filling a Memory with Defined Contents .....	13-7
13.3.5.1.5	Initializing SRAMs .....	13-7
13.3.5.1.6	Reading a Single Memory Location .....	13-9
13.3.5.1.7	Writing to a Single Memory Location .....	13-9
13.3.6	Resets and Clocks in the MTU, SSH & SRAM .....	13-9
13.3.6.1	Clock Domains .....	13-9
13.3.6.2	Reset Domains .....	13-10
13.3.6.2.1	Alarm Handling after Reset .....	13-10
13.3.7	SRAM Addressing and Scrambling .....	13-11

13.3.8	MBIST Algorithms .....	13-11
13.3.8.1	Non-Destructive Test (NDT) .....	13-11
13.4	Registers .....	13-14
13.4.1	Registers Overview .....	13-16
13.4.2	Register Description .....	13-18
13.4.2.1	System Registers .....	13-18
13.4.2.2	MTU Configuration Registers .....	13-20
13.4.2.3	SRAM Support Hardware (SSH) Registers .....	13-24
13.5	Safety Measures .....	13-41
13.5.1	Safety Features .....	13-41
13.5.1.1	SRAM Error Detection & Correction (EDC/ECC) .....	13-41
13.5.1.2	Address Error Monitor .....	13-42
13.5.1.3	SRAM Mux Factor .....	13-42
13.5.1.4	Error Tracking Registers .....	13-42
13.5.1.5	Safety Flip-Flops .....	13-43
13.5.2	Safety Notifications .....	13-43
13.5.2.1	Alarm Handling .....	13-45
13.5.2.1.1	Alarms after startup .....	13-46
13.5.2.1.2	Diagnostics .....	13-46
13.5.2.1.3	Error Mapping .....	13-47
13.5.2.1.4	Error Injection and Alarm Triggering .....	13-48
13.6	Revision History .....	13-49
<b>14</b>	<b>General Purpose I/O Ports and Peripheral I/O Lines (Ports) .....</b>	<b>14-1</b>
14.1	Feature List .....	14-1
14.2	Overview .....	14-1
14.3	Functional Description .....	14-3
14.3.1	System Connectivity of Ports .....	14-3
14.4	Registers .....	14-5
14.4.1	Module Identification Register .....	14-7
14.4.2	Port Input/Output Control Registers .....	14-8
14.4.3	Pad Driver Mode Register .....	14-12
14.4.4	LVDS Pad Control Register .....	14-15
14.4.5	Pin Function Decision Control Register .....	14-18
14.4.6	Pin Controller Select Register .....	14-19
14.4.7	Port Output Register .....	14-20
14.4.8	Port Output Modification Register .....	14-21
14.4.9	Port Output Modification Set Register .....	14-22
14.4.10	Port Output Modification Set Registers .....	14-23
14.4.11	Port Output Modification Clear Register .....	14-26
14.4.12	Port Output Modification Clear Registers .....	14-27
14.4.13	Emergency Stop Register .....	14-30
14.4.14	Port Input Register .....	14-31
14.4.15	Access Protection Registers .....	14-32
14.5	Revision History .....	14-34
<b>15</b>	<b>Safety Management Unit (SMU) .....</b>	<b>15-1</b>
15.1	Feature List .....	15-2
15.2	Overview .....	15-2
15.2.1	Architecture .....	15-4
15.2.2	SMU_core .....	15-5

15.2.3	SMU_stdby .....	15-5
15.3	Functional Description .....	15-6
15.3.1	SMU_core .....	15-6
15.3.1.1	Reset Types .....	15-6
15.3.1.2	Interfaces Overview .....	15-6
15.3.1.2.1	Interfaces to SCU .....	15-6
15.3.1.2.2	Interfaces to the Interrupt Router .....	15-7
15.3.1.2.3	Interface to the Ports (ErrorPin) .....	15-7
15.3.1.2.4	Interface to the Register Monitor .....	15-10
15.3.1.2.5	Interface to SMU_stdby .....	15-11
15.3.1.3	SMU_core Integration Guidelines .....	15-13
15.3.1.4	Alarm Mapping .....	15-14
15.3.1.4.1	SMU_core Internal Alarms .....	15-14
15.3.1.5	Alarm Handling .....	15-15
15.3.1.5.1	Alarm protocol .....	15-15
15.3.1.5.2	Alarm Configuration .....	15-15
15.3.1.5.3	Alarm operation .....	15-15
15.3.1.5.4	Alarm Status Registers .....	15-17
15.3.1.5.5	Alarm Diagnosis Registers .....	15-17
15.3.1.5.6	Port Emergency Stop .....	15-17
15.3.1.5.7	Recovery Timer .....	15-18
15.3.1.5.8	Watchdog Alarms .....	15-18
15.3.1.6	SMU_core Control Interface .....	15-19
15.3.1.7	SMU_core State Machine .....	15-22
15.3.1.8	Fault Signaling Protocol (FSP) .....	15-24
15.3.1.8.1	Introduction .....	15-24
15.3.1.8.2	Bi-stable fault signaling protocol .....	15-25
15.3.1.8.3	Timed dual rail .....	15-26
15.3.1.8.4	Time switching protocol .....	15-26
15.3.1.8.5	FSP Fault State .....	15-28
15.3.1.8.6	FSP and SMU_core START State .....	15-29
15.3.1.9	OCDS Trigger Bus (OTGB) Interface .....	15-30
15.3.1.10	Register Properties .....	15-31
15.3.1.10.1	Register Write Protection .....	15-31
15.3.1.10.2	Safety Flip-flops .....	15-31
15.3.2	SMU_stdby .....	15-32
15.3.2.1	Reset Types .....	15-32
15.3.2.2	Interfaces Overview .....	15-32
15.3.2.2.1	Interface to the Pads (ErrorPin) .....	15-32
15.3.2.3	Alarm Mapping .....	15-34
15.3.2.3.1	SMU_stdby Internal Alarms .....	15-34
15.3.2.4	Alarm Handling .....	15-35
15.3.2.4.1	Alarm protocol .....	15-35
15.3.2.4.2	Alarm Configuration .....	15-35
15.3.2.5	Register Properties .....	15-36
15.3.2.5.1	Register Write Protection .....	15-36
15.3.2.5.2	Safety Flip-flops .....	15-36
15.3.2.6	SMU_stdby Built-In Self Test .....	15-36
15.3.3	Interdependency Between SMU_core and SMU_stdby .....	15-36
15.4	Registers .....	15-36

15.4.1	SMU_core Module Registers .....	15-37
15.4.1.1	System Registers description .....	15-40
15.4.1.2	SMU_core Configuration Registers .....	15-44
15.4.1.3	SMU_core Alarm Configuration Registers .....	15-64
15.4.1.4	SMU_core Alarm Configuration Registers (Fault Signaling Protocol) .....	15-64
15.4.1.5	SMU_core Alarm Status Registers .....	15-66
15.4.1.6	SMU_core Alarm Diagnosis Registers .....	15-66
15.4.1.7	SMU_core Special Safety Registers: Register Monitor .....	15-67
15.4.2	SMU_stby Module Registers .....	15-69
15.4.2.1	SMU_stby Command Register .....	15-69
15.4.2.2	SMU_stby Alarm Configuration Register (Fault Signaling Protocol) .....	15-71
15.4.2.3	SMU_stby Alarm Status Register .....	15-73
15.4.2.4	SMU_stby BIST Control Register .....	15-74
15.4.2.5	SMU_stby BIST Status Register .....	15-75
15.5	Revision History .....	15-76
<b>16</b>	<b>Interrupt Router (IR)</b> .....	<b>16-1</b>
16.1	Feature List .....	16-1
16.2	Delta to TC2xx .....	16-1
16.3	Overview .....	16-2
16.4	Service Request Nodes (SRN) .....	16-3
16.4.1	Service Request Control Registers .....	16-3
16.4.1.1	General Service Request Control Register Format .....	16-4
16.4.1.1.1	Service Request Control Register (SRC) .....	16-4
16.4.1.2	Changing the SRN configuration .....	16-7
16.4.1.3	Protection of the SRC Registers .....	16-7
16.4.1.4	Request Set and Clear Bits (SETR, CLRR) .....	16-9
16.4.1.5	Enable Bit (SRE) .....	16-9
16.4.1.6	Service Request Flag (SRR) .....	16-9
16.4.1.7	Type-Of-Service Control (TOS) .....	16-9
16.4.1.8	Service Request Priority Number (SRPN) .....	16-10
16.4.1.9	ECC Encoding (ECC) .....	16-11
16.4.1.10	Interrupt Trigger Overflow Bit (IOV) .....	16-11
16.4.1.11	Interrupt Trigger Overflow Clear Bit (IOVCLR) .....	16-12
16.4.1.12	SW Sticky Bit (SWS) .....	16-12
16.4.1.13	SW Sticky Clear Bit (SWSLCR) .....	16-12
16.5	Mapping of Module Interrupt Request Triggers to SRNs .....	16-12
16.5.1	SRC Index Number .....	16-13
16.5.2	Interrupts related to the Debug Reset .....	16-13
16.5.3	Timing characteristics of Service Request Trigger Signals .....	16-14
16.6	Interrupt Control Unit (ICU) .....	16-15
16.6.1	ICU Interface to ISP .....	16-15
16.6.2	ICU Control Registers .....	16-16
16.6.2.1	Latest Winning Service Request Register (LWSR) .....	16-16
16.6.2.2	Last Acknowledged Service Request Register (LASR) .....	16-17
16.6.2.3	Error Capture Register (ECR) .....	16-17
16.7	General Purpose Service Requests, Service Request Broadcast .....	16-19
16.7.1	General Purpose Service Requests (GPSRxy) .....	16-19
16.7.2	Service Request Broadcast Registers (SRBx) .....	16-20
16.7.3	Access protection of SRBx registers (ACCEN_SRBy) .....	16-20

16.8	System Registers .....	16-20
16.8.1	Write Protection of Interrupt Router registers .....	16-20
16.8.2	Kernel Reset Registers (KRST1/0, KRSTCLR) .....	16-22
16.8.3	Clock Control Register (CLC) .....	16-22
16.8.4	OCDS Control and Status Register (OCS) .....	16-22
16.9	Arbitration Process .....	16-22
16.9.1	Number of Clock Cycles per Arbitration Process .....	16-23
16.9.2	Service Request Valid .....	16-24
16.9.3	Service Request Enter .....	16-24
16.9.4	Service Request Acknowledge .....	16-24
16.9.5	Handling of detected ECC Errors .....	16-24
16.10	Usage of the Interrupt System .....	16-25
16.10.1	CPU to ICU Interface .....	16-25
16.10.2	DMA to ICU Interface .....	16-25
16.10.3	Software-Initiated Interrupts .....	16-25
16.10.4	External Interrupts .....	16-26
16.11	Use Case Examples .....	16-26
16.11.1	Use Case Example Interrupt Handler .....	16-26
16.12	Module Implementation .....	16-28
16.12.1	Characteristics of the Interrupt Router Module .....	16-28
16.13	Interrupt Router System and Module Registers .....	16-28
16.13.1	System and ICU Control Registers .....	16-31
16.14	OTGM Registers .....	16-34
16.14.1	Status and Control .....	16-35
16.14.2	IRQ MUX Control .....	16-36
16.14.3	Interrupt System Trace .....	16-38
16.14.4	MCDS Interface .....	16-39
16.15	Revision History .....	16-39
<b>17</b>	<b>Flexible CRC Engine (FCE)</b> .....	<b>17-1</b>
17.1	Feature List .....	17-1
17.2	Overview .....	17-3
17.2.1	Application Mapping .....	17-3
17.2.2	Block Diagram .....	17-3
17.3	Functional Description .....	17-5
17.3.1	Initialization .....	17-8
17.3.2	Basic Operation .....	17-8
17.3.3	Automatic Signature Check .....	17-9
17.3.4	Register protection and monitoring methods .....	17-11
17.3.5	Power, Reset and Clock .....	17-13
17.3.6	Properties of CRC code .....	17-14
17.3.7	Service Request Generation .....	17-14
17.4	Registers .....	17-15
17.4.1	System Registers description .....	17-19
17.4.2	FCE Common Registers .....	17-23
17.4.3	CRC Channel Control/Status Registers .....	17-25
17.5	Debug .....	17-30
17.6	IO Interfaces .....	17-31
17.7	Revision History .....	17-31
<b>18</b>	<b>Direct Memory Access (DMA)</b> .....	<b>18-1</b>

18.1	Feature List .....	18-3
18.2	Overview .....	18-4
18.3	Functional Description .....	18-5
18.3.1	Configuration Interface .....	18-5
18.3.2	Resource Partitions .....	18-5
18.3.2.1	Access Enable .....	18-5
18.3.2.2	DMA Moves .....	18-5
18.3.2.3	DMA RP Error Interrupt Service Request .....	18-5
18.3.3	DMA Channels .....	18-5
18.3.3.1	DMA Channel Request Control .....	18-5
18.3.3.1.1	DMA Channel States .....	18-6
18.3.3.1.2	Reset Request Only After Transaction (RROAT) .....	18-7
18.3.3.2	DMA Software Request .....	18-7
18.3.3.3	DMA Hardware Request .....	18-8
18.3.3.4	Combined DMA Software Request and DMA Hardware Request .....	18-10
18.3.3.5	DMA Daisy Chain Request .....	18-10
18.3.3.6	DMA Channel Transaction Request Lost Interrupt Service Request .....	18-11
18.3.3.7	DMA Service Requests .....	18-11
18.3.3.8	DMA Request Arbitration .....	18-12
18.3.3.9	DMA Channel Reset .....	18-12
18.3.3.10	DMA Channel Halt .....	18-13
18.3.4	DMA Random Access Memory .....	18-15
18.3.4.1	DMA Channel Operation .....	18-16
18.3.4.2	DMA Channel Updates .....	18-18
18.3.4.2.1	Shadow Operations .....	18-18
18.3.4.2.2	Double Buffering Operations .....	18-19
18.3.4.3	DMA Channel Reconfiguration .....	18-19
18.3.4.4	Move Operation .....	18-20
18.3.4.4.1	Address Generation .....	18-20
18.3.4.4.2	Address Calculation Examples .....	18-21
18.3.4.4.3	Circular Buffer .....	18-22
18.3.4.4.4	Address Alignment .....	18-23
18.3.4.4.5	Address Counter .....	18-25
18.3.4.4.6	DMA Address Checksum .....	18-25
18.3.4.4.7	DMA Channel Interrupt Service Request .....	18-25
18.3.4.4.8	DMA Channel Transfer Interrupt Service Request .....	18-25
18.3.4.4.9	DMA Channel Pattern Match Interrupt Service Request .....	18-26
18.3.4.4.10	DMA Channel Wrap Buffer Interrupt Service Request .....	18-27
18.3.4.5	Shadow Operation .....	18-27
18.3.4.5.1	Application of Shadow Operation .....	18-27
18.3.4.5.2	Shadowed Address Register .....	18-27
18.3.4.5.3	Read Only Mode .....	18-27
18.3.4.5.4	Direct Write Mode .....	18-28
18.3.4.5.5	Error Conditions .....	18-29
18.3.4.5.6	Transfer Count Update .....	18-29
18.3.4.6	DMA Timestamp .....	18-30
18.3.4.6.1	Generation of DMA Timestamp .....	18-30
18.3.4.6.2	Appendage of DMA Timestamp to Non Destination Circular Buffer .....	18-30
18.3.4.6.3	Appendage of DMA Timestamp to Destination Circular Buffer .....	18-31
18.3.4.6.4	Application of DMA Timestamp .....	18-32

18.3.4.7	Pattern Detection .....	18-32
18.3.4.7.1	Pattern Compare Logic .....	18-32
18.3.4.7.2	Pattern Detection for 8-bit Channel Data Width .....	18-33
18.3.4.7.3	Pattern Detection for 16-bit Channel Data Width .....	18-35
18.3.4.7.4	Pattern Detection for 32-bit Channel Data Width .....	18-37
18.3.4.8	Double Buffering Operations .....	18-38
18.3.4.8.1	DMA Double Source Buffering .....	18-38
18.3.4.8.2	DMA Double Destination Buffering .....	18-38
18.3.4.8.3	Size of Buffer .....	18-39
18.3.4.8.4	Buffer Switch .....	18-39
18.3.4.8.5	Software Switch .....	18-40
18.3.4.8.6	Automatic Hardware Switch .....	18-40
18.3.4.8.7	Application of Double Buffering .....	18-40
18.3.4.9	Linked List Operations .....	18-43
18.3.4.9.1	DMA Auto Start Request .....	18-43
18.3.4.9.2	Non Linked List Operation .....	18-43
18.3.4.9.3	Last DMA Transaction .....	18-43
18.3.4.9.4	Circular Linked List Operations .....	18-43
18.3.4.9.5	DMA Linked List (DMALL) .....	18-44
18.3.4.9.6	Accumulated Linked List (ACLL) .....	18-44
18.3.4.9.7	Safe Linked List (SAFLL) .....	18-44
18.3.4.9.8	Conditional Linked List (CONLL) .....	18-47
18.3.4.10	DMA Data Checksum .....	18-49
18.3.4.11	DMARAM Initialization .....	18-51
18.3.5	Move Engine .....	18-51
18.3.5.1	ME Read Buffer .....	18-51
18.3.5.1.1	DMA Address Checksum .....	18-51
18.3.5.2	ME Error Conditions .....	18-51
18.3.5.3	Error Interrupt Service Request .....	18-52
18.3.5.3.1	DMARAM Integrity Error Interrupt Service Request .....	18-52
18.3.5.3.2	Source and Destination Error Interrupt Service Request .....	18-52
18.3.5.3.3	Linked List Operation TCS Error Interrupt Service Request .....	18-53
18.3.5.3.4	SAFLL DMA Address Checksum Error Interrupt Service Request .....	18-53
18.3.6	DMA On Chip Bus .....	18-53
18.3.6.1	DMA On Chip Bus Switch .....	18-53
18.3.6.1.1	SRI Master Interfaces .....	18-54
18.3.6.1.2	DMA On Chip Bus Switch Arbitration .....	18-54
18.3.6.2	On Chip Bus Master Interfaces .....	18-54
18.3.6.3	SRI Alarm .....	18-55
18.3.7	Power Modes .....	18-55
18.3.7.1	Sleep Mode .....	18-55
18.4	Register .....	18-57
18.4.1	Safety Flip-Flops .....	18-59
18.4.2	Register .....	18-60
18.4.3	DMA Resource Partition Registers .....	18-63
18.4.4	DMA Channel Registers .....	18-65
18.4.5	DMARAM Channel Registers .....	18-69
18.4.6	ME Registers .....	18-80
18.5	Debug .....	18-94
18.5.1	DMA Channel Suspend .....	18-94

18.5.2	Software Activation of DMA Channel Interrupt Service Requests .....	18-95
18.5.3	Software Activation of DMA RP Error Interrupt Service Requests .....	18-95
18.5.4	OCDS Trigger Bus (OTGB) Interface .....	18-95
18.5.5	MCDS Trace Interface .....	18-96
18.6	Use Cases .....	18-98
18.6.1	Move Operation .....	18-98
18.6.1.1	Step Description to Initialize and Trigger a DMA Transaction .....	18-98
18.6.2	Error Handler .....	18-98
18.6.3	Data Communication .....	18-99
18.7	Revision History .....	18-99
<b>19</b>	<b>Signal Processing Unit (SPU)</b> .....	<b>19-1</b>
19.1	Feature List .....	19-1
19.2	Overview .....	19-2
19.2.1	Glossary of Terms .....	19-2
19.2.2	Processing Flow .....	19-3
19.2.3	Use Case examples .....	19-3
19.2.3.1	SPU Configuration 1 .....	19-4
19.2.3.2	SPU Configuration 2 .....	19-5
19.2.3.3	SPU Configuration 3 .....	19-6
19.2.3.4	Thresholding .....	19-7
19.2.3.4.1	User Defined Thresholding .....	19-7
19.2.3.4.2	CFAR Based Thresholding Methods .....	19-7
19.2.3.5	Using Pre-acquisition Ramps .....	19-7
19.2.4	Elevation support .....	19-8
19.2.5	Phase demodulation .....	19-9
19.2.5.1	Alternate chirp demodulation .....	19-9
19.2.5.2	Static demodulation .....	19-9
19.2.5.3	HW optimisation for demodulation .....	19-9
19.2.6	Debugging .....	19-9
19.2.7	Execution flow .....	19-10
19.2.7.1	Execution flow for 4 Antennae .....	19-10
19.2.8	Memory mapping .....	19-10
19.2.8.1	Principle .....	19-11
19.2.8.2	Memory mapping for FFTs .....	19-11
19.2.8.3	Data Block Construction Control .....	19-11
19.2.8.3.1	Default Memory Mode .....	19-11
19.2.8.3.2	Integration Mode .....	19-12
19.2.8.4	Bandwidth Optimised Integration Mode .....	19-12
19.2.8.5	Memory mapping for other SPU results .....	19-13
19.2.8.6	Data sharing between SPU .....	19-13
19.2.8.6.1	Complex memory map via DMA reconfiguration .....	19-13
19.2.8.7	Example Memory mapping for 4 antenna and 16 bit operands .....	19-13
19.2.8.8	Data Read Order for 2nd stage FFT with 4 antennae and 16 bit operands .....	19-15
19.2.8.8.1	Data Mapping for 2nd Stage FFT results with 16bit Operands and 4 Antennae .....	19-16
19.2.8.8.2	Data Read Sequence in Integration Mode .....	19-17
19.2.8.8.3	Data Mapping for Integration Mode 2nd Stage FFT results with 16bit Operands and 4 Antennae .....	19-18
19.2.8.9	3rd stage FFT .....	19-20
19.2.8.10	Memory mapping for 4 antenna and 32bit operands .....	19-21

19.2.8.11	Data Mapping for 2nd Stage FFT with 4 antennae and 32 bit operands .....	19-22
19.2.8.12	Data Organisation in Integration Mode .....	19-24
19.2.8.13	Memory mapping for 3 antenna and 16bit operands .....	19-26
19.2.8.14	Data Read Sequence for 2nd Stage FFT with 3 antennae and 16 bit operands .....	19-27
19.2.8.15	Data Mapping for 2nd Stage FFT results with 16bit Operands and 3 Antennae .....	19-28
19.2.8.16	Data Read Order in Integration Mode .....	19-29
19.2.8.17	Data Mapping for 2nd Stage FFT results with 16bit Operands and 3 Antennae .....	19-30
19.2.8.18	Memory mapping for 6 antenna and 16bit operands .....	19-31
19.2.8.19	Data Read Order for 2nd Stage FFT with 6 antennae and 16 bit data .....	19-32
19.2.8.20	Data Mapping for 2nd Stage FFT results with 16bit Operands and 6 Antennae .....	19-33
19.2.8.21	Data Read Sequence in Integration Mode .....	19-34
19.2.8.22	Integration Mode Data Mapping for 2nd Stage FFT results with 16bit Operands and 6 Antennae ... 19-35	
19.3	Functional Description .....	19-36
19.3.1	Input DMA Engine .....	19-36
19.3.1.1	Load ADC data from the RIF .....	19-36
19.3.1.1.1	Principles of Operation (ADC IF) .....	19-36
19.3.1.1.2	Split Processing .....	19-37
19.3.1.2	Load from Radar Memory .....	19-37
19.3.1.2.1	Principles of Operation (Radar Memory) .....	19-39
19.3.1.2.2	Case 1: "Bin Offset" is set to Sample Size .....	19-42
19.3.1.2.3	Case 2: "Inner Loop Offset" is set to Sample Size .....	19-42
19.3.1.2.4	Case 3: "Outer Loop Offset" is set to Sample Size .....	19-42
19.3.1.2.5	Bandwidth Optimisation for Default Processing Mode .....	19-43
19.3.1.2.6	Bandwidth Optimisation Integration Processing Mode.....	19-43
19.3.1.3	Partial-acquisition Counter .....	19-44
19.3.1.4	FFT Data to Antenna Mapping .....	19-46
19.3.1.5	Reading Power Data From Radar Memory .....	19-46
19.3.1.6	Reading 16 bit Real Data From Radar Memory .....	19-46
19.3.1.6.1	Buffer Memory Switching for 16 bit real data .....	19-47
19.3.1.7	Data Storage in Buffer Memory .....	19-47
19.3.2	Streaming Processor 1 .....	19-47
19.3.2.1	Double Pass Mode .....	19-47
19.3.2.1.1	Double Pass Switch Mode .....	19-48
19.3.2.1.2	Window Parameter Switch .....	19-48
19.3.2.2	Data Loader Unit .....	19-48
19.3.2.2.1	Data Reformatting .....	19-49
19.3.2.2.2	Integration Mode Bandwidth Optimisation .....	19-49
19.3.2.2.3	Overview of Data Truncation and Padding .....	19-49
19.3.2.3	MATH1 Unit .....	19-50
19.3.2.3.1	Truncation .....	19-51
19.3.2.3.2	Windowing .....	19-51
19.3.2.3.3	Phase Shift .....	19-51
19.3.2.3.4	Padding .....	19-52
19.3.2.4	FFT Accelerator .....	19-52
19.3.3	Data Unloader .....	19-53
19.3.3.1	Power Histogram .....	19-53
19.3.3.2	Statistical Information .....	19-56
19.3.4	Streaming Processor 2, The Output Data Processor .....	19-56
19.3.4.1	Streaming Processor 2 Data Fetch from Buffer Memory .....	19-56

19.3.4.2	In Place FFT .....	19-57
19.3.4.2.1	Restrictions .....	19-57
19.3.5	MATH2 Unit .....	19-57
19.3.5.1	Pre-Processing Units .....	19-60
19.3.5.1.1	Linear Power Calculation .....	19-60
19.3.5.1.2	$\log_2$ Power Calculation .....	19-60
19.3.5.1.3	Magnitude Approximation .....	19-61
19.3.5.1.4	Saturating Truncation .....	19-61
19.3.5.2	Local Maximum Detection Unit .....	19-61
19.3.5.3	FFT Data Output Path .....	19-62
19.3.5.3.1	Scalar Addition .....	19-63
19.3.5.3.2	Complex Rescaling .....	19-63
19.3.5.3.3	Bin Rejection Unit .....	19-63
19.3.5.3.4	Half Precision Floating Point Format .....	19-65
19.3.5.4	Non-Coherent Integration .....	19-66
19.3.5.5	Constant False Alarm Rate Module .....	19-67
19.3.5.5.1	Inline Mode .....	19-67
19.3.5.5.2	Off-line Mode .....	19-67
19.3.5.5.3	CFAR Engine Architecture .....	19-67
19.3.5.5.4	CFAR Engine Data Format .....	19-68
19.3.5.5.5	CFAR Module Configuration .....	19-68
19.3.5.5.6	GOS-CFAR Engine .....	19-69
19.3.5.5.7	CA-CFAR Engine .....	19-70
19.3.5.5.8	CFAR Engine Configuration Restrictions .....	19-71
19.3.5.5.9	CFAR spectrum extension .....	19-71
19.3.5.5.10	Operation of Spectrum Extension .....	19-75
19.3.5.6	Summation Sideband Operations .....	19-75
19.3.5.6.1	Organisation .....	19-76
19.3.5.7	Statistical Information .....	19-76
19.3.6	Output DMA Engine .....	19-77
19.3.6.1	Output DMA Engine Channels .....	19-77
19.3.6.2	Data Cube Organisation and Size after processing ADC data .....	19-78
19.3.7	Radar sequencer .....	19-79
19.3.7.1	General Configuration .....	19-80
19.3.7.2	Radar sequencer start / stop .....	19-81
19.3.7.3	Synchronized Radar sequencer Start .....	19-82
19.3.7.4	Configuration / Reconfiguration .....	19-82
19.3.7.5	Linked Lists Organization .....	19-82
19.3.7.6	CPU monitoring during run time .....	19-82
19.3.7.7	Interrupts .....	19-83
19.3.8	Streaming Processor 1, Buffer RAM Switching Behaviour .....	19-84
19.3.8.1	Data Source is Radar Interface .....	19-84
19.3.8.2	Data Source is Radar Memory .....	19-84
19.3.9	Configuration Memory .....	19-84
19.3.9.1	Safety/Security .....	19-85
19.3.9.2	Configuration Register Data Format .....	19-85
19.3.9.2.1	Loading Configuration Settings .....	19-85
19.3.9.3	Window Data Format .....	19-85
19.3.9.4	Configuration Memory Usage Restrictions .....	19-85
19.4	Registers .....	19-86

19.4.1	Register Description .....	19-88
19.5	Debug .....	19-177
19.5.1	Trace Format .....	19-177
19.5.2	Debugger events .....	19-178
19.6	Safety Measures .....	19-179
19.6.1	Hardware Safety Mechanisms .....	19-179
19.6.1.1	Lockstep .....	19-179
19.6.1.1.1	Data Comparison Lockstep .....	19-179
19.6.1.2	Register CRC .....	19-179
19.6.1.3	RIF Interface CRC Check .....	19-180
19.6.1.4	Bypass Data CRC Check .....	19-180
19.6.1.5	Radar Memory Control Signal Redundancy .....	19-180
19.6.1.6	Radar Memory Tile Access Error .....	19-181
19.6.1.7	Radar Memory Read Data ECC .....	19-181
19.6.1.8	RAM ECC .....	19-181
19.6.1.9	RAM Address Signature ROM .....	19-181
19.6.1.10	Access Enable .....	19-181
19.6.2	Software Based Safety Mechanisms .....	19-181
19.6.2.1	Software Based Self Test .....	19-181
19.6.2.2	SPU Execution Time Check .....	19-182
19.6.2.3	Configuration Memory Content Check .....	19-182
19.6.3	Hardware Functionality Supporting Software Safety Mechanisms .....	19-182
19.6.3.1	Monitor Counters .....	19-182
19.6.3.2	Monitor CRC Units .....	19-182
19.6.3.3	Redundant Control Logic .....	19-183
19.6.4	SMU events .....	19-183
19.6.5	Safety assumptions .....	19-184
19.6.5.1	Safety assumptions and safety goals .....	19-184
19.6.5.1.1	Case1 = Radar with 1 Radar SPU only for low end applications .....	19-184
19.6.5.1.2	Case2 = Radar with 2 Radar SPUs only for mid to high end Radar .....	19-185
19.6.5.1.3	Case3 = Radar + sensor fusion with 2 Radar SPUs only for mid to high end Radar .....	19-185
19.6.5.2	Radar Application assumptions .....	19-185
19.6.5.2.1	Case 1: A decision is never taken on single ADC acquisition .....	19-185
19.6.5.2.2	Case 2: A decision is never taken on a single FFT computation (1st stage and 2nd stage FFTs) ... 19-185	
19.6.5.2.3	Case 3: FFT peaks are never isolated .....	19-185
19.7	Use Cases .....	19-187
19.7.1	Use of FFT Clock Division (CTRL.DIV) .....	19-187
19.7.2	In Place FFT .....	19-187
19.7.3	In Place FFT with ADC Data .....	19-187
19.7.4	ODM/MATH2 Dataset Sizes .....	19-188
19.7.5	In-line CFAR .....	19-188
19.7.6	CFAR Configuration .....	19-188
19.7.7	Non-Coherent Integration .....	19-188
19.7.8	FFT Data Output Path .....	19-188
19.7.9	Using SUMCTRL.SUMMODE=SUMANT with unconstrained ILR value .....	19-188
19.7.10	Kernel Reset and Lockstep .....	19-189
19.7.11	Supported Clocking Modes .....	19-189
19.7.12	RAM Initialization .....	19-189
19.7.13	Two SPU Instances Writing to the Same Radar Memory Tile .....	19-190

19.7.14	Performance of the SPU .....	19-190
19.7.14.1	Input Data Manager Performance, RIF Data .....	19-191
19.7.14.2	Input Data Manager Performance, EMEM Data .....	19-191
19.7.14.3	LOADER/FFT/UNLOADER Performance .....	19-192
19.7.14.4	MATH2/ODM Performance .....	19-192
19.7.14.5	Effect of “Double Pass” on Performance .....	19-193
19.8	I/O Interfaces .....	19-193
19.9	Revision History .....	19-196
<b>20</b>	<b>Signal Processing Unit 2 (SPU2) .....</b>	<b>20-1</b>
20.1	Feature List .....	20-1
20.2	Overview .....	20-2
20.3	Functional Description .....	20-2
20.3.1	Input DMA Engine .....	20-2
20.3.1.1	Load ADC data from the RIF .....	20-3
20.3.1.1.1	Principles of Operation (ADC IF) .....	20-3
20.3.1.1.2	Bypass Function .....	20-4
20.3.1.2	Load from Radar Memory .....	20-5
20.3.1.2.1	Principles of Operation (Radar Memory) .....	20-7
20.3.1.2.2	Case 1: “Bin Offset” is used to Iterate Across Adjacent Samples .....	20-10
20.3.1.2.3	Case 2: “Inner Loop Offset” is set to Iterate Across Adjacent Samples .....	20-11
20.3.1.2.4	Case 3: “Outer Loop Offset” is set to Iterate Across Adjacent Samples .....	20-11
20.3.1.2.5	Bandwidth Optimisation for Default Processing Mode .....	20-12
20.3.1.2.6	Bandwidth Optimisation Integration Processing Mode .....	20-12
20.3.1.3	Partial-acquisition Counter .....	20-13
20.3.1.4	FFT Data to Antenna Mapping .....	20-14
20.3.1.5	Reading Power Data From Radar Memory .....	20-15
20.3.1.6	Reading 16 bit Real Data From Radar Memory .....	20-15
20.3.1.6.1	Buffer Memory Switching for 16 bit real data .....	20-15
20.3.1.7	Reading Half Precision Floating Point Data From Radar Memory .....	20-15
20.3.1.8	Reading Compressed Complex Data From Radar Memory .....	20-15
20.3.1.9	Data Storage in Buffer Memory .....	20-16
20.3.2	Streaming Processor 1 .....	20-16
20.3.2.1	Double Pass Mode .....	20-16
20.3.2.1.1	Double Pass Switch Mode .....	20-17
20.3.2.1.2	Window Parameter Switch .....	20-17
20.3.2.2	Data Loader Unit .....	20-17
20.3.2.2.1	Data Reformatting for MATH0 .....	20-17
20.3.2.2.2	Data Reformatting for MATH1 .....	20-17
20.3.2.3	MATH0 Unit .....	20-19
20.3.2.3.1	DC Offset Removal .....	20-21
20.3.2.3.2	Transient Removal .....	20-21
20.3.2.3.3	Interference Detection .....	20-22
20.3.2.3.4	Ramp Buffer .....	20-24
20.3.2.3.5	Filtering .....	20-24
20.3.2.3.6	Combine Select and Taper .....	20-25
20.3.2.3.7	Output to Radar Memory .....	20-27
20.3.2.4	MATH1 Unit .....	20-28
20.3.2.4.1	Zero Insertion .....	20-29
20.3.2.4.2	Overview of Data Truncation and Padding .....	20-29

20.3.2.4.3	Truncation .....	20-30
20.3.2.4.4	Windowing .....	20-30
20.3.2.4.5	Phase Shift .....	20-30
20.3.2.4.6	Padding .....	20-31
20.3.2.5	FFT Accelerator .....	20-31
20.3.3	Data Unloader .....	20-32
20.3.3.1	Power Histogram .....	20-33
20.3.3.2	Statistical Information .....	20-35
20.3.4	Streaming Processor 2, The Output Data Processor .....	20-36
20.3.4.1	Streaming Processor 2 Data Fetch from Buffer Memory .....	20-36
20.3.4.2	In Place FFT .....	20-36
20.3.4.2.1	Restrictions .....	20-37
20.3.5	MATH2 Unit .....	20-37
20.3.5.1	Pre-Processing Units .....	20-40
20.3.5.1.1	Linear Power Calculation .....	20-40
20.3.5.1.2	$\log_2$ Power Calculation .....	20-40
20.3.5.1.3	Magnitude Approximation .....	20-41
20.3.5.1.4	Saturating Truncation .....	20-41
20.3.5.2	Local Maximum Detection Unit .....	20-41
20.3.5.3	FFT Data Output Path .....	20-42
20.3.5.3.1	Scalar Addition .....	20-43
20.3.5.3.2	Complex Rescaling .....	20-43
20.3.5.3.3	Bin Rejection Unit .....	20-43
20.3.5.3.4	Half Precision Floating Point Format .....	20-48
20.3.5.3.5	FFT Data Compression .....	20-48
20.3.5.4	Dual Integration Units .....	20-48
20.3.5.4.1	Digital Beam Forming .....	20-49
20.3.5.4.2	Non-Coherent Integration .....	20-50
20.3.5.5	Constant False Alarm Rate Module .....	20-51
20.3.5.5.1	CFAR Input Sources .....	20-51
20.3.5.5.2	CFAR Engine Architecture .....	20-51
20.3.5.5.3	CFAR Engine Data Format .....	20-52
20.3.5.5.4	CFAR Module Configuration .....	20-52
20.3.5.5.5	GOS-CFAR Engine .....	20-53
20.3.5.5.6	CA-CFAR Engine .....	20-54
20.3.5.5.7	CFAR Engine Configuration Restrictions .....	20-55
20.3.5.5.8	CFAR spectrum extension .....	20-56
20.3.5.5.9	Operation of Spectrum Extension .....	20-59
20.3.5.6	Statistical Information .....	20-59
20.3.6	Output DMA Engine .....	20-60
20.3.6.1	Output DMA Engine Channels .....	20-60
20.3.6.2	Data Cube Organisation and Size after processing ADC data .....	20-61
20.3.7	Radar sequencer .....	20-62
20.3.7.1	General Configuration .....	20-62
20.3.7.2	Radar sequencer start / stop .....	20-63
20.3.7.3	Synchronized Radar sequencer Start .....	20-65
20.3.7.4	Configuration / Reconfiguration .....	20-65
20.3.7.5	Linked Lists Organization .....	20-65
20.3.7.6	CPU monitoring during run time .....	20-65
20.3.7.7	Interrupts .....	20-65

20.3.8	Streaming Processor 1, Buffer RAM Switching Behaviour .....	20-66
20.3.8.1	Data Source is Radar Interface .....	20-67
20.3.8.2	Data Source is Radar Memory .....	20-67
20.3.9	Configuration Memory .....	20-67
20.3.9.1	Safety/Security .....	20-67
20.3.9.2	Configuration Register Data Format .....	20-67
20.3.9.2.1	Loading Configuration Settings .....	20-67
20.3.9.3	Window Data Format .....	20-68
20.3.9.4	Configuration Memory Usage Restrictions .....	20-68
20.4	Registers .....	20-69
20.4.1	Register Description .....	20-73
20.5	Debug .....	20-189
20.5.1	Trace Format .....	20-189
20.5.2	Debugger events .....	20-190
20.6	Safety Measures .....	20-191
20.6.1	Hardware Safety Mechanisms .....	20-191
20.6.1.1	Register CRC .....	20-191
20.6.1.2	RIF Interface CRC Check .....	20-191
20.6.1.3	RIF Interface CRC Check for Bypass .....	20-192
20.6.1.4	Bypass Data CRC Check .....	20-192
20.6.1.5	Radar Memory Control Signal Redundancy .....	20-192
20.6.1.6	Radar Memory Tile Access Error .....	20-192
20.6.1.7	Radar Memory Read Data ECC .....	20-193
20.6.1.8	RAM ECC .....	20-193
20.6.1.9	RAM Address Signature ROM .....	20-193
20.6.1.10	Access Enable .....	20-193
20.6.2	Software Based Safety Mechanisms .....	20-193
20.6.2.1	Software Based Self Test .....	20-193
20.6.2.2	TC3Ax SPU Execution Time Check .....	20-193
20.6.2.3	Configuration Memory Content Check .....	20-194
20.6.3	Hardware Functionality Supporting Software Safety Mechanisms .....	20-194
20.6.3.1	Monitor CRC Units .....	20-194
20.6.3.2	Redundant Control Logic .....	20-194
20.6.4	SMU events .....	20-195
20.6.5	Safety assumptions .....	20-195
20.6.5.1	Safety assumptions and safety goals .....	20-195
20.6.5.1.1	Case1 = Radar with 1 Radar TC3Ax SPU only for low end applications .....	20-196
20.6.5.1.2	Case2 = Radar with 2 Radar SPUs only for mid to high end Radar .....	20-196
20.6.5.1.3	Case3 = Radar + sensor fusion with 2 Radar SPUs only for mid to high end Radar .....	20-196
20.6.5.2	Radar Application assumptions .....	20-196
20.6.5.2.1	Case 1: A decision is never taken on single ADC acquisition .....	20-196
20.6.5.2.2	Case 2: A decision is never taken on a single FFT computation (1st stage and 2nd stage FFTs) ...	20-196
20.6.5.2.3	Case 3: FFT peaks are never isolated .....	20-196
20.7	Use Cases .....	20-198
20.7.1	Using Fractional Clock Division (AUXCTRL.DIVF) .....	20-198
20.8	I/O Interfaces .....	20-198
20.9	Revision History .....	20-199
<b>21</b>	<b>Bit Manager (BITMGR)</b> .....	<b>21-1</b>

21.1	Feature List .....	21-1
21.2	Overview .....	21-2
21.2.1	Glossary of Terms .....	21-2
21.2.2	Functional Overview .....	21-3
21.3	Functional Description .....	21-4
21.3.1	Array operand format .....	21-4
21.3.2	Operations .....	21-6
21.3.2.1	Mode 0, Dual Array Overlay .....	21-6
21.3.2.2	Mode1, Sub-array overlay .....	21-7
21.3.2.3	Mode2, Output Replication .....	21-9
21.3.3	Target Label List Generation .....	21-10
21.3.4	Function triggers and function chaining .....	21-11
21.3.4.1	Independent operation of functions .....	21-11
21.3.4.2	Function chaining .....	21-11
21.3.4.3	SW Trigger .....	21-12
21.3.4.4	HW Trigger .....	21-12
21.3.5	Run mode .....	21-12
21.3.5.1	One-Shot trigger .....	21-12
21.3.5.2	Batch mode trigger .....	21-12
21.3.6	Interrupts .....	21-13
21.4	Process pipeline .....	21-14
21.5	Registers .....	21-16
21.5.1	Register Description .....	21-17
21.6	Debug .....	21-52
21.6.1	Trace .....	21-52
21.6.2	Debugger events .....	21-52
21.7	Safety Measures .....	21-53
21.7.1	Hardware Safety Mechanisms .....	21-53
21.7.1.1	Functional logic redundancy .....	21-53
21.7.1.2	Register CRC .....	21-53
21.7.1.3	Radar Memory Control Signal Redundancy .....	21-53
21.7.1.4	Radar Memory Tile Access Error .....	21-54
21.7.1.5	Radar Memory Read Data ECC .....	21-54
21.7.1.6	Private buffers based on RAM .....	21-54
21.7.1.7	Access Enable .....	21-54
21.7.2	Software Based Safety Mechanisms .....	21-54
21.7.3	Hardware Functionality Supporting Software Safety Mechanisms .....	21-54
21.7.3.1	Monitor CRC Units .....	21-55
21.7.3.2	Redundant Control Logic .....	21-55
21.7.4	SMU events .....	21-55
21.8	Use Cases .....	21-56
21.9	I/O Interfaces .....	21-59
21.10	Revision History .....	21-60
<b>22</b>	<b>SPU Lockstep Comparator (SPULCKSTP) .....</b>	<b>22-1</b>
22.1	Feature List .....	22-1
22.2	Overview .....	22-2
22.3	Functional Description .....	22-3
22.3.1	SPU Lockstep Control .....	22-3
22.3.2	SPU Lockstep Monitoring .....	22-3

22.3.3	Lockstep Self Test .....	22-4
22.3.4	Functional Redundancy .....	22-6
22.3.5	Lockstep Failure Signalling Test .....	22-6
22.4	Registers .....	22-7
22.4.1	Details of SPULCKSTP Registers .....	22-7
22.5	Use Cases .....	22-14
22.5.1	Conditions of Use .....	22-15
22.5.2	Set Up .....	22-15
22.5.2.1	Specific Setup for Full Lockstep .....	22-15
22.5.3	SPU Triggering .....	22-15
22.5.4	Expected Use Cases .....	22-15
22.6	Revision History .....	22-17
<b>23</b>	<b>Extension Memory (EMEM) .....</b>	<b>23-1</b>
23.1	Feature List .....	23-2
23.2	Overview .....	23-3
23.3	Functional Description .....	23-3
23.3.1	Isolation Logic .....	23-3
23.3.2	EMEM Modes .....	23-3
23.3.2.1	Locked Mode .....	23-4
23.3.2.2	Standby Locked Mode .....	23-4
23.3.2.3	Changing the EMEM Mode .....	23-4
23.3.3	Tile Modes .....	23-5
23.3.3.1	Application Mode .....	23-6
23.3.3.2	MCDS Mode .....	23-6
23.3.3.3	Tool Mode .....	23-6
23.3.3.4	Accessing Tiles in Different Modes .....	23-7
23.3.4	Address Map .....	23-7
23.3.4.1	Address View .....	23-8
23.3.4.2	XTM Addressing .....	23-8
23.3.4.2.1	MCDS Mode .....	23-9
23.3.4.2.2	Tool Mode .....	23-9
23.3.5	EMEM Module SRAM .....	23-9
23.3.5.1	SRAM Initialization .....	23-9
23.3.5.2	Memory Integrity Check .....	23-10
23.3.5.3	RAM Alarm .....	23-10
23.3.6	SRI Interface .....	23-10
23.3.6.1	Register Protection .....	23-10
23.3.6.2	Memory Protection .....	23-10
23.3.6.3	Memory Disabled .....	23-11
23.3.6.4	True and Inverted Logic .....	23-11
23.3.6.5	Error Detection and Signalling .....	23-11
23.3.6.5.1	Access Enable Violation .....	23-12
23.3.6.5.2	SRI Access Address Phase Error .....	23-12
23.3.6.5.3	SRI Write Access Data Phase Error .....	23-12
23.3.6.5.4	SRI Write Access to SRAM Error .....	23-12
23.3.6.5.5	SRI Read Access to SRAM Error .....	23-12
23.3.6.5.6	True and Inverted Logic Error .....	23-13
23.3.6.5.7	Internal Data Transfer ECC Error .....	23-13
23.3.6.6	Control Redundancy .....	23-14

23.3.6.6.1	Control Redundancy Test .....	23-15
23.3.6.6.2	Consistency Check .....	23-15
23.3.7	SEP Interface .....	23-15
23.3.7.1	TC39xED SEP Accesses to EMEM Tiles .....	23-15
23.3.7.2	TC35x SEP Accesses to EMEM Tiles .....	23-15
23.3.7.3	TC3Ax SEP Accesses to EMEM Tiles .....	23-16
23.3.7.4	TC33xED SEP Accesses to EMEM Tiles .....	23-16
23.3.7.5	SEP Error .....	23-16
23.3.7.6	SEP ECC Error .....	23-16
23.3.7.6.1	SEP Write Access .....	23-17
23.3.7.6.2	SEP Read Access .....	23-17
23.3.7.7	SPU Full Lockstep .....	23-17
23.3.8	Reset Control .....	23-17
23.3.9	Clock Control .....	23-17
23.4	Registers .....	23-17
23.4.1	EMEM Core Register Description .....	23-18
23.4.2	EMEM Module Register Description .....	23-25
23.4.2.1	EMEM Module General Registers .....	23-26
23.4.2.2	EMEM Module SRAM Protection Registers .....	23-30
23.4.3	EMEM Module RAM .....	23-34
23.4.4	EMEM XTM RAM .....	23-35
23.5	Revision History .....	23-36
<b>24</b>	<b>Radar Interface (RIF) .....</b>	<b>24-1</b>
24.1	Feature List .....	24-1
24.2	Overview .....	24-2
24.3	Functional Description .....	24-4
24.3.1	External Serial Interface (ESI) .....	24-4
24.3.2	Internal Parallel Interface (IPI) .....	24-5
24.3.3	Quad Processing Unit .....	24-7
24.3.4	Default CRC Scheme .....	24-7
24.3.5	Alternative CRC Scheme .....	24-9
24.3.5.1	Byte Swapping .....	24-9
24.3.6	CRC as a Safety Mechanism .....	24-9
24.3.7	Data Formatting Unit (DFU) .....	24-10
24.3.8	FIFO and Lane Management (FLM) .....	24-12
24.3.8.1	FLM Operating Modes .....	24-12
24.3.8.2	RIF Internal Lockstep and SPU CRC .....	24-13
24.3.8.3	Real and Complex Sampling .....	24-15
24.3.8.4	Multi Lane Real Sampling .....	24-16
24.3.8.5	Multiplex Complex Mode .....	24-16
24.3.9	Data Memory Interface (DMI) .....	24-18
24.3.9.1	Data Format of the Memory Interface .....	24-18
24.3.10	Radar State Machine (RSM) .....	24-19
24.3.11	External ADC Use-Case .....	24-19
24.3.11.1	Frame Watchdog .....	24-21
24.3.11.2	On-Chip Signal Delay Calibration .....	24-23
24.3.11.3	On-chip Signal Delay Calibration Sequence .....	24-24
24.3.11.4	Waveforms Required to Perform On-Chip Signal Delay Calibration .....	24-24
24.3.11.5	Delay Adjustment During Calibration .....	24-25

24.3.11.6	Skew Measurement During Calibration .....	24-25
24.3.11.7	Reference Skew Value and Error Limits .....	24-27
24.3.11.8	RAMP1 Signal .....	24-27
24.3.12	Internal ADCs Use-Case .....	24-28
24.3.13	Frequency Domains .....	24-28
24.3.13.1	Synchronization of two RIF Modules .....	24-29
24.3.13.2	Interrupts .....	24-31
24.3.14	OCDS Trigger Sets .....	24-32
24.3.15	Register CRC .....	24-33
24.3.16	Operating Modes .....	24-34
24.3.16.1	Sleep Mode .....	24-34
24.3.16.2	OCDS Suspend Mode .....	24-34
24.3.17	Module Implementation .....	24-34
24.3.17.1	ID Registers .....	24-34
24.3.17.2	Implementation Details .....	24-34
24.3.17.3	On-Chip Connections .....	24-35
24.3.17.3.1	Connections to the internal ADCs .....	24-35
24.3.17.3.2	RAMP1 Connections .....	24-36
24.4	Registers .....	24-37
24.4.1	Kernel Registers .....	24-39
24.4.2	BPI_FPI Registers .....	24-68
24.5	IO Interfaces .....	24-74
24.6	Revision History .....	24-75
<b>25</b>	<b>High Speed Pulse Density Modulation Module (HSPDM) .....</b>	<b>25-1</b>
25.1	Feature List .....	25-1
25.2	Functional Description .....	25-2
25.2.1	HSPDM Modes of Operation .....	25-2
25.2.1.1	Shift Register Generated Bit-Stream .....	25-2
25.2.1.2	Delta-sigma Modulator Generated Bit-Stream with the CIC filter and the Compactor enabled	25-3
25.2.1.3	Delta-sigma Modulator Generated Bit-Stream with the CIC filter and the Compactor disabled	25-6
25.2.2	HSPDM clocking and EVADC trigger generation .....	25-6
25.2.2.1	Internal Timer Module (ITM) .....	25-6
25.2.2.2	ADC Trigger Generation .....	25-6
25.2.3	Pad Asymmetry Compensation (PAC) .....	25-7
25.2.4	SRAM and Data Management .....	25-9
25.2.4.1	RAM Buffer Manager (RAMBM) .....	25-10
25.2.4.2	MUTE Signal Generation .....	25-11
25.2.4.3	Interrupts .....	25-13
25.2.4.4	Starting and Stopping the Bit-Streaming .....	25-13
25.2.4.5	Hardware Run Feature .....	25-14
25.3	Registers .....	25-15
25.3.1	Kernel Registers .....	25-18
25.3.2	BPI_FPI Registers .....	25-28
25.4	IO Interfaces .....	25-34
25.5	Revision History .....	25-34
<b>26</b>	<b>Camera and ADC Interface (CIF) .....</b>	<b>26-1</b>
26.1	Feature List .....	26-1
26.2	Overview .....	26-1
26.2.1	Introduction .....	26-2

26.2.1.1	Camera and ADC Interface Functional Overview .....	26-2
26.2.1.2	Camera and ADC Interface Block Diagram .....	26-2
26.2.2	Camera and ADC Interface Functional Specification .....	26-3
26.2.2.1	Target Applications .....	26-3
26.2.2.1.1	Camera Interface Example .....	26-3
26.2.2.1.2	Connecting External ADC .....	26-4
26.3	Functional Description .....	26-8
26.3.1	Sub Module ISP .....	26-8
26.3.2	Sub Module Security Watchdog .....	26-9
26.3.3	Sub Module Y/C-Split .....	26-10
26.3.4	Sub Module JPEG Encoder .....	26-10
26.3.5	Sub Module Linear Downscaler .....	26-11
26.3.6	Sub Module Extra Path Units .....	26-12
26.3.7	Debug Path .....	26-14
26.3.8	Sub Module Memory Interface (MI) .....	26-16
26.3.8.1	Write to EMEM .....	26-19
26.3.9	BBB Master Interface .....	26-20
26.3.10	BBB Slave Interface .....	26-21
26.3.11	Control Unit .....	26-21
26.3.12	Shadow Registers .....	26-21
26.3.13	CIF Module Integration and BPI Adapter .....	26-22
26.3.13.1	BPI_SPB Module Registers .....	26-22
26.3.13.1.1	System Registers .....	26-23
26.3.14	CIF Programming Hints .....	26-29
26.3.14.1	Configuration and Shadow Registers .....	26-29
26.3.14.2	General Setup for Operation .....	26-30
26.3.14.3	Start-Stop Programming .....	26-31
26.3.14.3.1	Data capturing controlled by the ISP .....	26-31
26.3.14.4	Abort of Processing .....	26-31
26.3.14.4.1	Frame Skip .....	26-31
26.3.14.4.2	Handling Picture Size Error .....	26-32
26.3.14.5	Interrupt Handling .....	26-32
26.3.14.5.1	ISP Events .....	26-32
26.3.14.5.2	Memory Interface (MI) Events .....	26-34
26.3.14.6	Reset Handling .....	26-35
26.3.14.7	Programming Guide .....	26-35
26.3.14.7.1	ISP Programming .....	26-35
26.3.14.7.2	Memory Interface Programming .....	26-41
26.3.14.7.3	Getting Started - First steps for startup .....	26-45
26.3.14.8	Use Case Description .....	26-47
26.3.14.8.1	Data Transfer .....	26-48
26.3.14.8.2	Viewfinder Mode .....	26-49
26.3.14.8.3	Still Image Capture .....	26-50
26.3.14.9	Power Management .....	26-51
26.3.14.10	Basics on Configuration Access .....	26-51
26.4	Registers .....	26-52
26.4.1	CIF Control Registers .....	26-53
26.4.1.1	CIF Clock Control Registers .....	26-64
26.4.1.2	CIF Custom Registers .....	26-65
26.4.1.3	CIF Internal Control Registers .....	26-66

26.4.2	ISP Programming Registers .....	26-69
26.4.2.1	ISP Control Registers .....	26-69
26.4.2.2	ISP Acquisition Registers .....	26-71
26.4.2.3	ISP Output Control Registers .....	26-75
26.4.2.4	ISP Interrupt Control Registers .....	26-80
26.4.2.5	Miscellaneous ISP Registers .....	26-85
26.4.3	Linear Downscaler Programming Registers .....	26-88
26.4.3.1	Linear Downscaler Configuration Registers .....	26-88
26.4.4	Memory Interface Programming Registers .....	26-90
26.4.4.1	Memory Interface Control Registers .....	26-90
26.4.4.2	Memory Interface Shadow Registers .....	26-103
26.4.4.3	Memory Interface Interrupt Registers .....	26-109
26.4.5	JPEG Encoder Programming Registers .....	26-116
26.4.5.1	JPEG Encoder Control Registers .....	26-116
26.4.5.2	JPEG Encoder Interrupt Registers .....	26-129
26.4.6	Security Watchdog Programming Registers .....	26-135
26.4.6.1	Watchdog Configuration Registers .....	26-135
26.4.6.2	Watchdog Interrupt Registers .....	26-137
26.4.7	ISP Image Stabilization Registers .....	26-141
26.4.7.1	Image Stabilization Control Registers .....	26-141
26.4.7.2	Image Stabilization Shadow Registers .....	26-145
26.4.8	Extra Path Programming Registers .....	26-148
26.4.8.1	Extra Path Error Registers .....	26-148
26.4.8.2	Memory Interface Extra Path Interrupt Registers .....	26-150
26.4.8.3	Memory Interface Extra Path Control Registers .....	26-157
26.4.8.4	Extra Path Memory Interface Shadow Registers .....	26-164
26.4.8.5	Extra Path Image Cropping Control Registers .....	26-167
26.4.8.6	Extra Path Image Cropping Shadow Registers .....	26-172
26.4.9	Debug Path Programming Registers .....	26-174
26.4.9.1	Debug Path Control Registers .....	26-174
26.4.9.2	Debug Path Status Registers .....	26-176
26.4.9.3	Debug Path User Defined Symbols Registers .....	26-177
26.5	IO Interfaces .....	26-177
26.6	Revision History .....	26-178

**Revision history .....****RevisionHistory-1**

## Introduction

# 1 Introduction

This User's Manual describes the Infineon AURIX™ TC3xx Platform, a range of 32-bit multicore microcontrollers based on the Infineon TriCore Architecture.

## 1.1 About this Document

This document is designed to be read primarily by potential users of a AURIX™ TC3xx Platform product who need a detailed description of the products and their functional units.

### 1.1.1 Related Documentations

A complete description of the TriCore architecture is found in the document entitled “TriCore Architecture Manual”. The architecture of the TriCore is described separately this way because of the configurable nature of the TriCore specification: different versions of the architecture may contain a different mix of systems components. The TriCore architecture, however, remains constant across all derivative designs in order to preserve compatibility.

This User's Manual should be read in conjunction with the “TriCore Architecture Manual”.

### 1.1.2 Text Conventions

This document uses the following text conventions for named components:

- Functional units are given in plain UPPER CASE. For example: “The QSPI supports full-duplex and half-duplex synchronous communication”.
- Pins using negative logic are indicated by an overline. For example: “The external reset pin, ESR0, has a dual function”.
- Bit fields and bits in registers are in general referenced as “Module\_Register name.Bit field” or “Module\_Register name.Bit”. For example: “The Current CPU Priority Number bit field CPU\_ICR.CCPN is cleared”. Most of the register names contain a module name prefix, separated by an underscore character “\_” from the actual register name (for example, “QSPI0\_GLOBALCON”, where “QSPI0” is the module name prefix, and “GLOBALCON” is the kernel register name). In chapters describing the kernels of the peripheral modules, the registers are mainly referenced with their kernel register names. The peripheral module implementation sections mainly refer to the actual register names with module prefixes.
- Variables used to describe sets of processing units or registers appear in mixed upper and lower cases. For example, register name “MOFCRn” refers to multiple “MOFCR” registers with variable n. The bounds of the variables are always given where the register expression is first used (for example, “n = 0-255”), and are repeated as needed in the rest of the text.
- The default radix is decimal. Hexadecimal constants are suffixed with a subscript letter “H”, as in: 100<sub>H</sub>. Binary constants are suffixed with a subscript letter “B”, as in: 111<sub>B</sub>.
- When the extent of register fields, groups register bits, or groups of pins are collectively named in the body of the document, they are represented as “NAME[A:B]”, which defines a range for the named group from A to B. Individual bits, signals, or pins are given as “NAME[C]” where the range of the variable C is given in the text. For example: CFG[2:0] and SRPN[0].
- Units are abbreviated as follows:
  - **MHz** = Megahertz
  - **µs** = Microseconds
  - **kBaud, kbit/s** = 1000 characters/bits per second
  - **MBaud, Mbit/s** = 1000000 characters/bits per second
  - **Kbyte, KB** = 1024 bytes of memory

## Introduction

- **Mbyte**, MB= 1048576 bytes of memory  
In general, the k prefix scales a unit by 1000 whereas the K prefix scales a unit by 1024. Hence, the Kbyte unit scales the expression preceding it by 1024. The kBaud unit scales the expression preceding it by 1000. The M prefix scales by 1000000 or 1048576, and  $\mu$  scales by .000001. For example, 1 Kbyte is 1024 bytes, 1 Mbyte is  $1024 \times 1024$  bytes, 1 kBaud/kbit are 1000 characters/bits per second, 1 MBaud/Mbit are 1000000 characters/bits per second, and 1 MHz is 1000000 Hz.
- Data format quantities are defined as follows:
  - **Byte** = 8-bit quantity
  - **Half-word** = 16-bit quantity
  - **Word** = 32-bit quantity
  - **Double-word** = 64-bit quantity

### 1.1.3 Family Specification and Appendix

The User's Manual is split into a "Family" part (named "AURIXTC3XX\_\*) and an "Appendix" per silicon.

The Family part describes each module used in this family of devices. It is intended as reference for the human reader to understand functionality and register set of these modules. Additionally it contains a family wide address map, a feature set table and all block diagrams.

The Appendix describes differences of a module implementation for a certain device series. Its content is usually supplied by tools (e.g. compiler, debugger, configuration tools). These differences are usually device connectivity (connections to pins or other modules), the Register Address Space (i.e. a list of implemented modules and their address ranges) and specific register implementations like bit fields that are only functional in certain devices. Registers that are identical to the Family part contain a cross reference to the Family documentation.

When bus interfaces are shared between modules (e.g. for SCU, CCU, PMS) the Appendix is also shared to allow an accurate listing of all registers in a certain address range.

### 1.1.4 Register and Memory Address Documentation

Some modules enable through their bus interface access to memory ranges and registers. In these cases the "Register Address Space" table shows memories differently, for example:

**Table 1 Register Address Space - LMU\_DAM**

Module	Base Address	End Address	Note
(DAM0)	90400000 <sub>H</sub>	9040FFFF <sub>H</sub>	DAM RAM Access cached address space
	B0400000 <sub>H</sub>	B040FFFF <sub>H</sub>	DAM RAM Access non-cached address space
DAM0	F8500000 <sub>H</sub>	F8507FFF <sub>H</sub>	Special Function Register Address Space

In this example "LMU\_DAM" notifies the module name. In the first column "(DAM0)" and all following rows with empty Module field contain memory ranges accessible by this module instance DAM0. The row showing "DAM0" in the first column contains the address range definition of the DAM0 registers.

The following "Register Overview" table lists all registers with offset addresses based on the row containing "DAM0".

If the LMU\_DAM has multiple instances (e.g. DAM0, DAM1) then the table has further rows with "(DAM1)" for memories and "DAM1" as base address for the registers.

## Introduction

### 1.1.5 Reserved, Undefined, and Unimplemented Terminology

In tables where register bit fields are defined, the following conventions are used to indicate undefined and unimplemented function. Furthermore, types of bits and bit fields are defined using the abbreviations as shown in [Table 2](#).

**Table 2 Bit Function Terminology**

Function of Bits	Description
<b>Unimplemented, Reserved</b>	Register bit fields named <b>0</b> indicate unimplemented functions with the following behavior: <ul style="list-style-type: none"><li>• Reading these bit fields returns 0.</li><li>• These bit fields should be written with 0 if the bit field is defined as <b>r</b> or <b>rh</b>.</li><li>• These bit fields have to be written with 0 if the bit field is defined as <b>rw</b>.</li></ul> These bit fields are reserved. The detailed description of these bit fields can be found in the register descriptions.
<b>rw</b>	The bit or bit field can be read and written.
<b>rwh</b>	As <b>rw</b> , but bit or bit field can be also set or reset by hardware.
<b>r</b>	The bit or bit field can only be read (read-only).
<b>w, wX</b>	The bit or bit field can only be written (write-only). A read of this register will always give a default value back that is described in the register documentation.
<b>w0, w1</b>	The bit or bit field can only be written (write-only). A read of this register will always give the value 0 or 1 back.
<b>r0, r1</b>	The bit or bit field can only be read. The read value is 0 or 1.
<b>rh</b>	This bit or bit field can be modified by hardware (read-hardware, typical example: status flags). A read of this bit or bit field give the actual status of this bit or bit field back. Writing to this bit or bit field has no effect on the setting of this bit or bit field.
<b>s</b>	Bits with this attribute are “sticky” in one direction. If their reset value is once overwritten by software, they can be switched again into their reset state only by a reset operation. Software cannot switch this type of bit into its reset state by writing the register. This attribute can be combined to “rws” or “rwhs”.
<b>f</b>	Bits with this attribute are readable only when they are accessed by an instruction fetch. Normal data read operations will return other values.

### 1.1.6 Register Access Modes

Read and write access to registers and memory locations are sometimes restricted. In memory and register access tables, the terms as defined in [Table 3](#) are used.

In general, if an access type is not permitted under these rules (e.g. attempted write to R, attempted user mode access to SV, attempted access to E without Endinit, etc.) then a **Bus Error** will result, unless the access is also marked as nBE (or otherwise stated in the specific module chapter).

Other special access restrictions may apply in some modules. These will be described within the module chapters.

## Introduction

**Table 3 Access Terms**

Symbol	Description
BE	Always returns Bus Error (e.g. used on Write Access to indicate a read-only register)
CPUx	Access only by CPUx (identified by its bus master id).
CEy	Access only when CPUy ENDINIT is not active (SCU_WDTCPUyCON0.ENDINIT = 0)
E	Access only when any CEx is inactive (SCU_WDTCPUxCON0.ENDINIT = 0 for any CPUx), or SCU_EICON0.ENDINIT = 0
H	Access only from HSM Master (and Cerberus, if HSM Debug is enabled) when DFLASH1 is HSM Exclusive (DMU_SP_PROCONHSMCFG.HSMDX is set)
M	Marks module specific access condition which is described in the module's chapter
OEN	Access only when OCDS is enabled
P	Access only from Master x (when MOD_ACCEN0.ENx = 1)
P0	Access only from Master x (when MOD_ACCEN00.ENx = 1) <sup>1)</sup>
P1	Access only from Master x (when MOD_ACCEN10.ENx = 1) <sup>1)</sup>
Pr	Access only from Master x (when MOD_ACCENr0.ENx or MOD_ACCENr1.ENx = 1) <sup>2)</sup>
P00/P01	Access only from Master x (when MOD_ACCEN0/01.ENx = 1, i.e. r=0) <sup>2)</sup>
PW	Access only when correct Password
SE	Access only when Safety Endinit is inactive (SCU_WDTSCON0.ENDINIT = 0 or SCU_SEICON.ENDINIT = 0)
ST	Access only when startup (SSW) executes
SV	Access only for when Supervisor Mode is active on the bus.
TM	Access only when SCU test mode
U	Access only when User Mode is active on the bus.
32	Access only when 32-bit width
32,64	Access only when 32-bit or 64-bit width

1) Different definition in IR module, also defining "P1" and "P2".

2) Definitions used in DMA module to separate resource partitions "r".

**Table 4 Combined Access Conditions**

Symbol	Description
U, SV	U or SV
P, U, SV	P and (U or SV)
P, SV, E	P and SV and E
SV, SE	SV and SE
TM, ST	TM or ST

More complicated Access Conditions (e.g. write access depending on value of other register) described in text or separate "Access Mode Restrictions" tables.

**Introduction****Table 5 Other Register Annotations**

<b>Symbol</b>	<b>Description</b>
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.
nE	Indicates that no Error is generated when accessing this address or address range, even though the access is to an undefined address or address range. True for CPU accesses (MTCR/MFCR) to undefined addresses in the CSFR range.

**1.1.7 Register Reset Documentation**

From application point of view the register reset value is the initial content of the register upon application start. In addition to the documented hardware reset types (see headline “Reset Types” in chapter “Reset Control Unit (RCU)”) also the Startup Software can affect the initial content. Finally the resulting content may depend on conditions. Reset tables linked to the register definition convey this information.

In the following example “LVD Reset” and “Cold PORST” are hardware reset types and “After SSW execution” shows that the SSW overwrites this register with a different value:

**Table 6 Reset Values of <Name of Register>**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
LVD Reset	0059 7F4A <sub>H</sub>	
Cold PORST	0059 7F4A <sub>H</sub>	
After SSW execution	005C 834A <sub>H</sub>	

The following Reset Type definitions can be found in this documentation:

**Table 7 Defined Reset Types**

<b>Reset Type</b>	<b>Description</b>
Application Reset	see RCU chapter
PowerOn Reset	see RCU chapter
System Reset	see RCU chapter
Cold PowerOn Reset	see RCU chapter. Also abbreviated “Cold PORST”.
Warm PowerOn Reset	see RCU chapter. Also abbreviated “Warm PORST”.
Debug Reset	see RCU chapter. Also name “Debug Clear”.
LVD Reset	Low Voltage Detector reset, see PMS chapter.
Kernel Reset	Module internal hardware reset triggered with KRST0 and KRST1 registers. Also named “Module Reset”.
other name	Module specific reset type described in respective chapter usually, among them: <ul style="list-style-type: none"> <li>• IOClient Reset (OCDS)</li> <li>• MCDS Reset (MiniMCDS)</li> <li>• DPLL_RAM_INI.INIT... (GTM)</li> <li>• Generated Reset (SCR)</li> </ul>

---

## Introduction

**Table 7    Defined Reset Types (cont'd)**

<b>Reset Type</b>	<b>Description</b>
CFS Value	Indicating that CFS or UCB contain a reference value for this register. This value can be copied automatically by hardware or SSW into the register or application software has to perform the copying. Details are described in the respective chapter.
Default Flash	Indicating values in a Flash location (UCB or CFS) that are contained at delivery time.
After SSW execution	Register overwritten by SSW.

The “Reset Value” column may contain letter “-” and “X” to indicate bits or nibbles that are either not affected by this reset type or are changed to a value not known at design time (e.g. trimming value).

The “Note” column may describe further conditions when this particular value is applied.

---

## Introduction

### 1.1.8 Abbreviations and Acronyms

The following acronyms and terms are used in this document:

**Table 8 Abbreviations and Acronyms**

Acronym	Description
ADAS	Advanced Driver Assistance System
ADC	Analog-to-Digital Converter
ALU	Arithmetic and Logic Unit
ASCLIN	Asynchronous/Synchronous Serial Controller with LIN
BBB	Back Bone Bus
BCU	Bus Control Unit
BROM	Boot ROM & Test ROM
CAN	Controller Area Network
CIF	Camera (and ADC) Interface
CCU	Clock Control Unit
CPU	Central Processing Unit
CRC	Cyclic Redundancy Code
CSA	Context Save Area
CSFR	Core Special Function Register
CCU6	Capture Compare Unit 6
DAM	Default Application Memory
DAP	Device Access Port
DAS	Device Access Server
DPI	Direct Processor Interface (to Local Flash Bank)
DCACHE	Data Cache
DFLASH (or DF)	Data Flash Memory
DLMU	Direct-connected Local Memory Unit
DMA	Direct Memory Access
DMBI	Data Memory Bus Interface
DMI	Data Memory Interface
DMU	Data Memory Unit
DRLB	Data Read Line Buffer
DSPR	Data Scratchpad RAM
EBU	External Bus Interface
ECC	Error Correction Code
ED	Emulation Device
EDSADC	Enhanced Delta-Sigma Analog to Digital Converter
EVADC	Enhanced Versatile Analog-to-Digital Converter
EMI	Electro-Magnetic Interference
ERAY	Flexray Controller

---

## Introduction

**Table 8 Abbreviations and Acronyms (cont'd)**

<b>Acronym</b>	<b>Description</b>
EtherMAC	Ethernet Media Access Controller
EVR	Embedded Voltage Regulator
FCE	Flexible CRC Engine
FCOMP	Fast Comparator
FM-PLL	PLL with Frequency Modulation support
FPI	Flexible Peripheral Interconnect (Bus protocol)
FPU	Floating Point Unit
FSM	Finite State Machine
GPIO	General Purpose Input/Output
GPT12	General Purpose Timer 12
GTM	Generic Timer Module
HSM	Hardware Security Module
HSPDM	High Speed Pulse Density Modulator
HSSL	High Speed Serial Link
I2C	Inter-Integrated Circuit Controller
I/O	Input / Output
IOM	I/O Monitor Unit
IR	Interrupt Router
JTAG	Joint Test Action Group = IEEE1149.1
LMU	Local Bus Memory Unit
MBIST	Memory Build In Self Test
MMU	Memory Management Unit
MSB	Most Significant Bit
MSC	Micro Second Channel
MTU	Memory Test Unit
MCMCAN	CAN controller
NC	Not Connected
NMI	Non-Maskable Interrupt
NVM	Non Volatile Memory
OCDS	On-Chip Debug Support
OVRAM	Overlay Memory
PLL	Phase Locked Loop
PCACHE	Program Cache
PD	Production Device
PFI	Program Flash Interface
PFLASH (or PF)	Program Flash Memory
PMBI	Program Memory Bus Interface
PMI	Program Memory Interface

---

## Introduction

**Table 8 Abbreviations and Acronyms (cont'd)**

<b>Acronym</b>	<b>Description</b>
PMS	Power Management System
PSI5	Peripheral Sensor Interface
PSI5-S	Peripheral Sensor Interface with Serial Interface to Phy
PSPR	Program Scratchpad RAM
QSPI	Queued SPI Controller
RAM	Random Access Memory
RCU	Reset Control Unit
RIF	Radar Interface
RISC	Reduced Instruction Set Computing
SBCU	System Peripheral Bus Control Unit
SCU	System Control Unit
SCR	Standby Controller
SENT	Single Edge Nibble Transmission
SFR	Special Function Register
SMU	Safety Management Unit
SPB	System Peripheral Bus (based on FPI protocol)
SPU	Signal Processing Unit
SPD	Single Pin DAP
SPI	Synchronous Serial Controller
SRI	Shared Resource Interconnect
SRAM	Static Data Memory
SRN	Service Request Node
STM	System Timer
SWD	Supply Watchdog
TC1.6.2P	TriCore CPU TC1.6.2P
UCB	User Configuration Block
WDT	Watchdog Timer
XBar, XBar_SRI	Cross Bar Interconnect, based on the Shared Resource Interconnect protocol

---

## Introduction

### 1.2 System Architecture of the AURIX™ TC3xx Platform

The AURIX™ TC3xx Platform is a family of high-performance microcontrollers with multiple TriCore CPUs, program and data memories, buses, bus arbitration, interrupt system, DMA controller, and a powerful set of on-chip peripherals. The AURIX™ TC3xx Platform is designed to meet the needs of the most demanding embedded control systems applications where the competing issues of price/performance, real-time responsiveness, computational power, data bandwidth, and power consumption are key design elements.

The AURIX™ TC3xx Platform offers several versatile on-chip peripheral units such as serial controllers, timer units, and analog-to-digital converters. Within the AURIX™ TC3xx Platform, all these peripheral units are connected to the TriCore CPUs / system via a System Peripheral Bus (SPB) and a Shared Resource Interconnect (SRI). A number of I/O lines on the AURIX™ TC3xx Platform ports are reserved for these peripheral units to communicate with the external world.

The TriCore processor architecture combines three powerful technologies within one processing unit, achieving new levels of power, speed, and economy for embedded applications:

- Reduced Instruction Set Computing (RISC) processor architecture
- Digital Signal Processing (DSP) operations and addressing modes
- On-chip memories and peripherals

DSP operations and addressing modes provide the computational power necessary to efficiently analyse complex real-world signals. The RISC load/store architecture provides high computational bandwidth with low system cost. On-chip memory and peripherals are designed to support even the most demanding high-bandwidth real-time embedded control-systems tasks.

Additional high-level features of the AURIX™ TC3xx Platform products include:

- Multicore Architecture
- Efficient memory organization: instruction and data scratch memories, caches, and local flash banks
- Serial communication interfaces – flexible synchronous and asynchronous modes
- Multiple channel DMA Controller – DMA operations and interrupt servicing
- Flexible interrupt system – configurable interrupt priorities and targets
- Hardware Security Module
- Flexible CRC Engine
- General-purpose timers
- High-performance on-chip buses
- On-chip debugging and emulation facilities
- Flexible interconnections to external components
- Flexible power-management

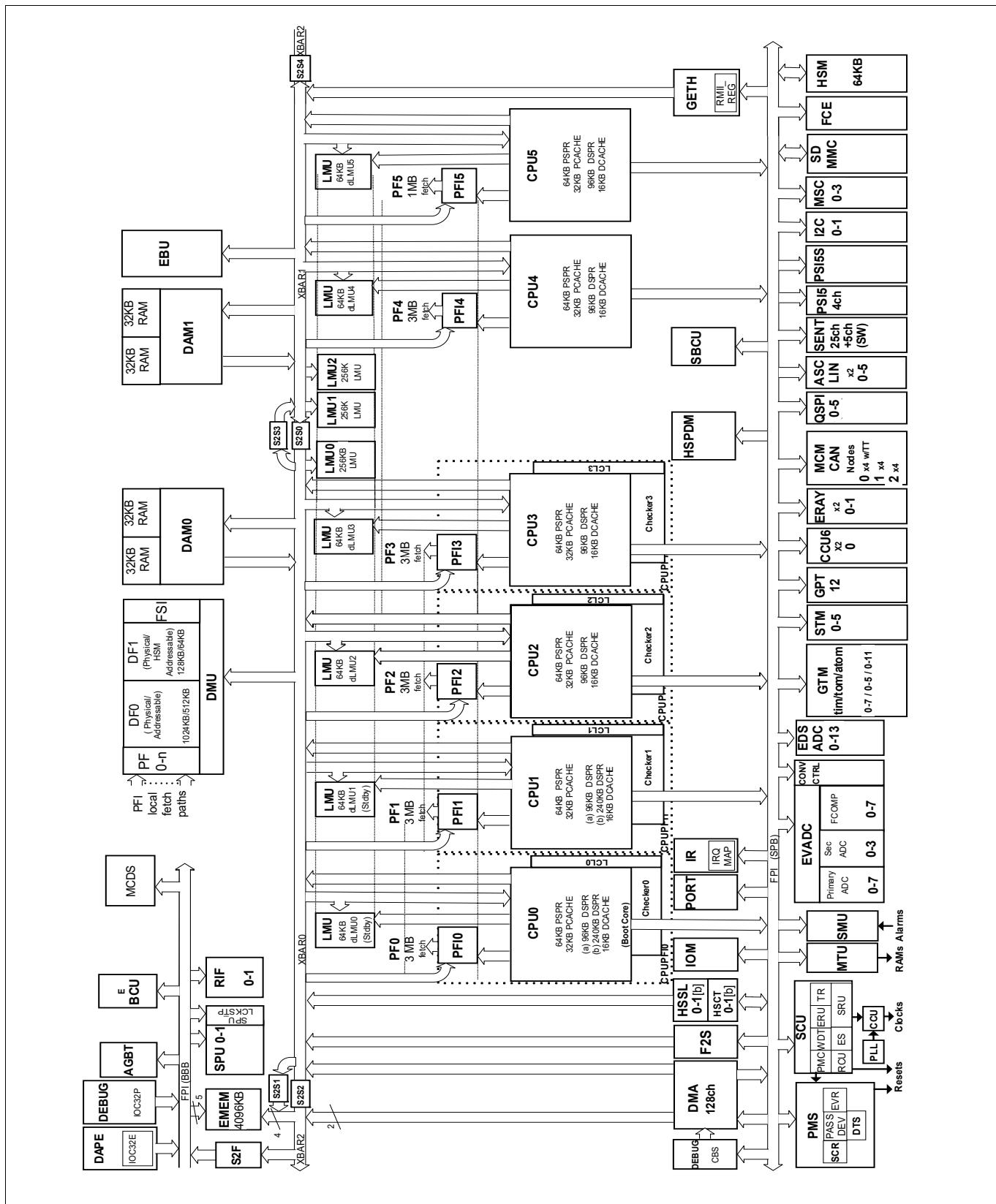
In addition, the following Advanced Driver Assistance System or Extended features are available in some products:

- Camera Interface
- Radar Interface
- Extended Memory
- ADAS Signal Processing Unit

## Introduction

## **1.2.1 AURIX™ TC3xx Platform High End – TC39x**

**Figure 1** shows the block diagram of the TC39x lead device.



**Figure 1** Block Diagram of TC39x

## Introduction

### 1.2.2 AURIX™ TC3xx Platform - Family Overview

The following pages show the block diagrams of the main platform devices which are based upon the TC3xx architecture.

The devices are:

- TC39x: Infineon internal name used in some tools and documentation “TC39xED”
- TC38x
- TC3Ex
- TC37xEXT: Infineon internal name used in some tools and documentation “TC37xED”
- TC37x
- TC36x
- TC35x
- TC3Ax
- TC33xEXT: Infineon internal name used in some tools and documentation “TC33xED”
- TC33x

For each of these devices an Appendix document is supplied. Devices of the same series might share one Datasheet.

The [Table 9](#) shows a feature overview of the AURIX™ TC3xx Platform family.

Due to pin limitations the usable functionality depends on the pinning of a certain package.

Note further that derivative products called “variants” are created from above device list with a reduced feature set, see [Section 1.2.3](#).

**Table 9 Platform Feature Overview**

Feature		TC33x TC33xEXT	TC35x TC3Ax	TC36x	TC37x TC37xEXT	TC38x TC3Ex	TC39x
CPUs	Type	TC1.6.2P					
	Cores / Checker Cores	TC33x: 1 / 1 TC33xEXT: 2 / 1	TC35x: 3 / 2 TC3Ax: 4 / 2	2 / 2	TC37x: 3 / 2 TC37xEXT: 3 / 3	4 / 2	6 / 4
	Max. Freq.	300 MHz	300 MHz	300 MHz	300 MHz	300 MHz	300 MHz
Cache per CPU	Program	32 KB	32 KB	32 KB	32 KB	32 KB	32 KB
	Data	16 KB	16 KB	16 KB	16 KB	16 KB	16 KB

## Introduction

**Table 9 Platform Feature Overview (cont'd)**

Feature		TC33x TC33xEXT	TC35x TC3Ax	TC36x	TC37x TC37xEXT	TC38x TC3Ex	TC39x
SRAM per CPU	PSPR	TC33x: 8 KB TC33xEXT: 32 KB in CPU0, 64 KB in CPU1	64 KB	32 KB	64 KB	64 KB	64 KB
	DSPR	TC33x: 192 KB in CPU0 TC33xEXT: 192 KB in CPU0, 96 KB in CPU1	TC35x: 240 KB in CPU0&1, 96 KB in other CPUs TC3Ax: 240 KB in CPU0,1&2, 96 KB in other CPUs	192 KB	240 KB in CPU0&1, 96 KB in other CPUs	240 KB in CPU0&1, 96 KB in other CPUs	240 KB in CPU0&1, 96 KB in other CPUs
	DLMU	TC33x: 8 KB in CPU0 TC33xEXT: 8 KB in CPU0, 64 KB in CPU1	64 KB	64 KB	64 KB	64 KB	64 KB
SRAM global	LMU	-	TC35x: 512 KB TC3Ax: -	-	-	TC38x: 128 KB TC3Ex: 256 KB	768 KB
	DAM	-	-	-	32 KB	64 KB	128 KB
Extension Memory (EMEM)	TCM	TC33x: - TC33xEXT: 1 MB	TC35x: 2 MB TC3Ax: 6 MB	-	TC37x: - TC37xEXT: 2 MB	-	2 MB
	XCM	-	-	-	TC37x: - TC37xEXT: 1 MB	-	2 MB
	XTM	TC33x: - TC33xEXT: 16 KB	16 KB	-	TC37x: - TC37xEXT: 16 KB	-	16 KB
Program Flash	Size	2 MB	4 MB	4 MB	6 MB	TC38x: 10 MB TC3Ex: 12 MB	16 MB
	Banks	1 x 2 MB	2 x 2 MB	2 x 2 MB	2 x 3 MB	TC38x: 3 x 3 MB, 1 x 1 MB TC3Ex: 4 x 3 MB	5 x 3 MB, 1 x 1 MB

## Introduction

**Table 9 Platform Feature Overview (cont'd)**

Feature		TC33x TC33xEXT	TC35x TC3Ax	TC36x	TC37x TC37xEXT	TC38x TC3Ex	TC39x
Data Flash	DF0 Size (single-ended)	128 KB	128 KB	128 KB	256 KB	TC38x: 512 KB TC3Ex: 1 MB	1 MB
	DF1 Size (single-ended)	128 KB	128 KB	128 KB	128 KB	128 KB	128 KB
DMA	Channels	64	64	64	128	128	128
	Move Engines	2	2	2	2	2	2
	Resource Partitions	4	4	4	4	4	4
CONVCTRL	Modules	1					
EVADC	Primary Groups/ Channels	TC33x: 2 / 16 TC33xEXT: 6 / 40	2 / 16	4 / 32	4 / 32	8 / 64 <sup>1)</sup>	8 / 64
	Secondary Groups/ Channels	TC33x: 2 / 28 TC33xEXT: 0 / 0	0 / 0	2 / 32	4 / 64	4 / 64	4 / 64
	Fast Compare Channels	0	0	2	4	4	8
EDSADC	Channels	0	0	4	6	10 <sup>2)</sup>	14

## Introduction

**Table 9 Platform Feature Overview (cont'd)**

Feature		TC33x TC33xEXT	TC35x TC3Ax	TC36x	TC37x TC37xEXT	TC38x TC3Ex	TC39x
GTM	Clusters	TC33x: 2 @ 200MHz TC33xEXT: 0	0	4 @ 200MHz	6 (5 @ 200MHz, 1 @ 100MHz)	9 (5 @ 200MHz, 4 @ 100MHz)	12 (5 @ 200MHz, 7 @ 100MHz)
	TIM (8 ch)	TC33x: 2 TC33xEXT: 0	0	3	6	7	8
	TOM (16 ch)	TC33x: 2 TC33xEXT: 0	0	2	3	5	6
	ATOM (8 ch)	TC33x: 1 TC33xEXT: 0	0	4	6	9	12
	MCS (8 ch)	0	0	3	5	7	10
	CMU / ICM	TC33x: 1 / 1 TC33xEXT: 0	0	1 / 1	1 / 1	1 / 1	1 / 1
	PSM	0	0	1	1	2	3
	TBU channels <sup>3)</sup>	TC33x: 3 (TBU0-2) TC33xEXT: 0	0	4 (TBU0-3)	4 (TBU0-3)	4 (TBU0-3)	4 (TBU0-3)
	SPE	TC33x: 2 TC33xEXT: 0	0	2	2	4	6
	CMP / MON	TC33x: 1/1 TC33xEXT: 0/0	0 / 0	1 / 1	1 / 1	1 / 1	1 / 1
	BRC / DPLL	TC33x: 1/0 TC33xEXT: 0/0	0 / 0	1 / 1	1 / 1	1 / 1	1 / 1
CDTM modules	CDTM modules	TC33x: 2 TC33xEXT: 0	0	4	5	6	7
	DTM modules	TC33x: 6 (4 on TOM, 2 on ATOM) TC33xEXT: 0	0	12 (4 on TOM, 8 on ATOM)	16 (6 on TOM, 10 on ATOM)	20 (8 on TOM, 12 on ATOM)	24 (10 on TOM, 14 on ATOM)
Timer	GPT12	1					
	CCU6	1 module with 2 kernels					
STM	Modules	TC33x: 1 TC33xEXT: 2	TC35x: 3 TC3Ax: 4	2	3	4	6
FlexRay	Modules	TC33x: 1 TC33xEXT: 0	TC35x: 1 TC3Ax: 0	1			
	Channels	TC33x: 2 TC33xEXT: 0	TC35x: 2 TC3Ax: 0	2	2 x 2		

## Introduction

**Table 9 Platform Feature Overview (cont'd)**

Feature		TC33x TC33xEXT	TC35x TC3Ax	TC36x	TC37x TC37xEXT	TC38x TC3Ex	TC39x
CAN	Modules	TC33x: 2 TC33xEXT: 1	TC35x: 2 TC3Ax: 1	2	TC37x: 2 TC37xEXT: 3	TC38x: 3 TC3Ex: 5	3
	Nodes	TC33x: 2 x 4 TC33xEXT: 1 x 4	TC35x: 2 x 4 TC3Ax: 1 x 4	2 x 4	TC37x: 2 x 4 TC37xEXT: 3 x 4	TC38x: 3 x 4 TC3Ex: 5 x 4	3 x 4
	of which support TT-CAN	0	0	1	1	1	1
QSPI	Modules	4	4	4	5	5	6
	HSIC Channels	2	2	0	0	0	2
ASCLIN	Modules	TC33x: 12 TC33xEXT: 6	4	12	12	24 <sup>4)</sup>	12
I2C	Interfaces	0	1	1	1	2	2
SENT	Channels	6	0	10	15	25 <sup>5)</sup>	25
PSI5	Channels	0	0	2	2	4	4
PSI5-S	Channels	0	0	1	1	1	1
HSSL	Channels	0	TC35x: 0 TC3Ax: 2	1	1	1	2
MSC	Modules	0	0	1	2	3 <sup>6)</sup>	4
EBU	External Bus	0	0	0	0	0	1
SDMMC	eMMC/SD Interface	TC33x: 0 TC33xEXT: 1	0	0	TC37x: 0 TC37xEXT: 1	TC38x: 0 TC3Ex: 1	1
Gigabit-Ethernet (10M/100M/1Gbit) <sup>7)</sup>	Modules	TC33x: 0 TC33xEXT: 1	1	1	TC37x: 1 TC37xEXT: 2	1	1
ASIL	Level	up to ASIL-D					
FCE	Modules	1					
Safety Support	SMU	yes					
	IOM	TC33x: yes TC33xEXT: no	TC35x: yes TC3Ax: no	yes			
SPU	Modules	TC33x: 0 TC33xEXT: 1	2 <sup>8)</sup>	0	0	0	2
SPU Lockstep Cmp.	Modules	no	TC35x: yes TC3Ax: no	no	no	no	yes
BITMGR0	Modules	no	TC35x: no TC3Ax: yes	no			
RIF	Modules	TC33x: 0 TC33xEXT: 1	2 <sup>9)</sup>	0	0	0	2

## Introduction

**Table 9 Platform Feature Overview (cont'd)**

Feature		TC33x TC33xEXT	TC35x TC3Ax	TC36x	TC37x TC37xEXT	TC38x TC3Ex	TC39x
CIF	Modules	0	0	0	TC37x: 0 TC37xEXT: 1	0	0
HSPDM	Modules	TC33x: 0 TC33xEXT: 1	TC35x: 1 TC3Ax: 0	0	0	0	1
Security	HSM+	1					
Debug	OCDS	yes					
	MCDS	TC33x: no TC33xEXT: yes ("light version")	yes ("light version")	no	TC37x: no TC37xEXT: yes	no	yes
	miniMCDS	no	no	no	TC37x: yes TC37xEXT: no	yes	no
	miniMCDS TRAM	-	-	-	TC37x: 8 KB TC37xEXT: -	8 KB	-
	AGBT <sup>10)</sup>	TC33x: no TC33xEXT: yes	yes	no	TC37x: no TC37xEXT: yes	no	yes
Low Power Features	Standby RAM	DLMU0	DLMU0&1	DLMU0&1	DLMU0&1	DLMU0&1	DLMU0&1
	SCR	yes					
Power Management System	PMS	TC33x: PMSLE <sup>11)</sup> TC33xEXT: PMS	PMS				

## Introduction

**Table 9 Platform Feature Overview (cont'd)**

Feature		TC33x TC33xEXT	TC35x TC3Ax	TC36x	TC37x TC37xEXT	TC38x TC3Ex	TC39x
Packages	Bare Die	no	no	yes	TC37x: yes TC37xEXT: yes <sup>12)</sup>	TC38x: yes TC3Ex: no	yes <sup>12)</sup>
	LFBGA-516	no	no	no	no	TC38x: yes TC3Ex: no	yes
	LFBGA-292	TC33x: yes TC33xEXT: no	no	yes	TC37x: yes TC37xEXT: yes	TC38x: yes TC3Ex: yes	yes
	LFBGA-292 with special ADAS pinning	TC33x: no TC33xEXT: yes	yes	no	no	no	yes
	LQFP-176 (0.5mm)	no	no	yes	TC37x: yes TC37xEXT: yes <sup>13)</sup>	no	no
	TQFP-144 (0.4mm)	TC33x: yes TC33xEXT: no	no	yes	no	no	no
	LQFP-144 (0.5mm)	no	no	yes	TC37x: no TC37xEXT: yes <sup>13)</sup>	no	no
	TQFP-100 (0.4mm)	TC33x: yes TC33xEXT: no	no	no	no	no	no
	TQFP-80	TC33x: yes TC33xEXT: no	no	no	no	no	no
	LFBGA-180	TC33x: yes TC33xEXT: no	no	yes	no	no	no
	LFBGA-180 with special ADAS pinning	TC33x: no TC33xEXT: yes	TC35x: yes TC3Ax: no	no	no	no	no
	LFBGA-216 with special ADAS pinning	TC33x: no TC33xEXT: no	TC35x: no TC3Ax: yes	no	no	no	no

- 1) In TC3Ex due to pinning constraints only 5 primary groups are usable.
- 2) In TC3Ex due to pinning constraints only 6 channels are usable.
- 3) TBU3 has special purpose as angle clock.
- 4) ASCLIN instances ASCLIN12 to ASCLIN23 are usable only for LIN and UART communication. SPI is not usable as this function is not connected to pads.
- 5) In TC3Ex due to pinning constraints only 20 channels are usable.
- 6) In TC3Ex due to pinning constraints only MSC0 and MSC1 are usable.
- 7) Note: depending on the package not all pins for 1Gbit might be available.

---

## Introduction

- 8) In TC3Ax, SPU2 is used.
- 9) In TC3Ax, 600 Mbps / Lane is implemented.
- 10) Only in feature package “E”, i.e. designated emulation devices and some feature package “T” devices for development. And in certain feature package “A” devices for tracing in laboratory. Not available in productive devices. Availability depends also on the package.
- 11) Power Management System for Low-End.
- 12) For TC37xEXT and TC39x only available for Feature package “E” devices, i.e. designated Emulation devices. For these bare die is available but only in VEES (Very Early Engineering Sample) quality.
- 13) Only for Feature package “E” devices, i.e. designated Emulation devices. For these this package is available but not qualified.

## Introduction

### 1.2.2.1 TC38x Block Diagram

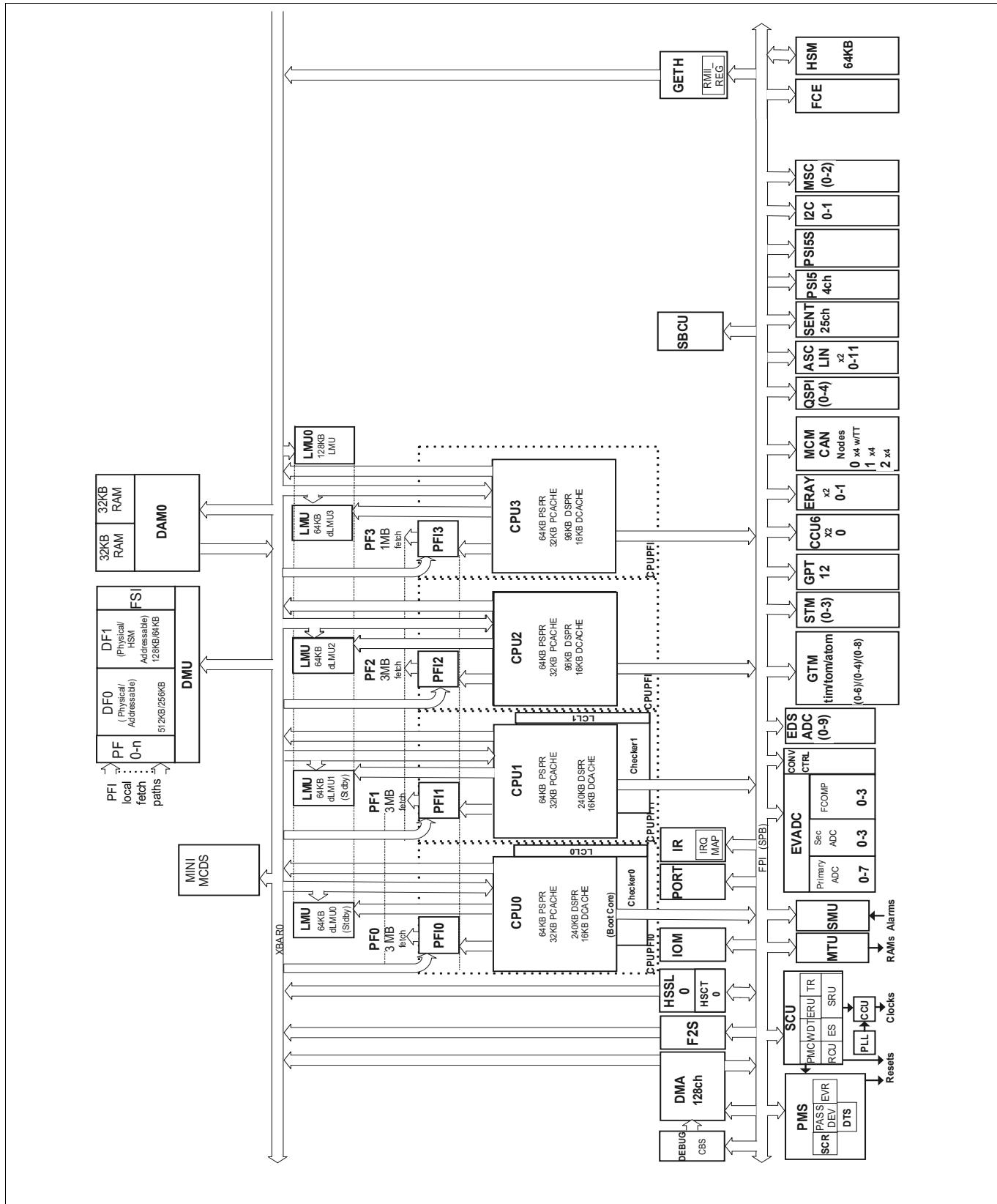
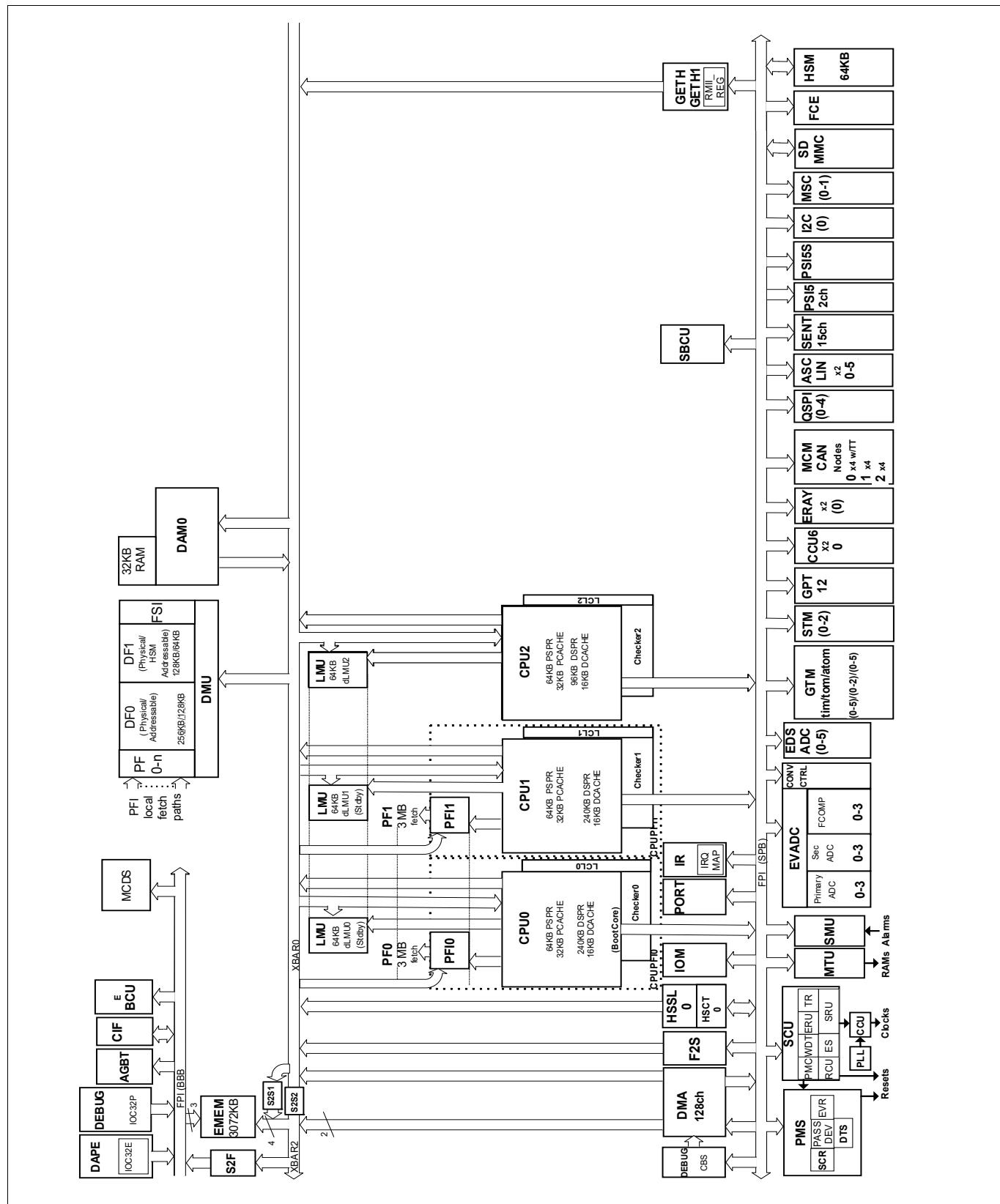


Figure 2 Block Diagram of TC38x

## Introduction

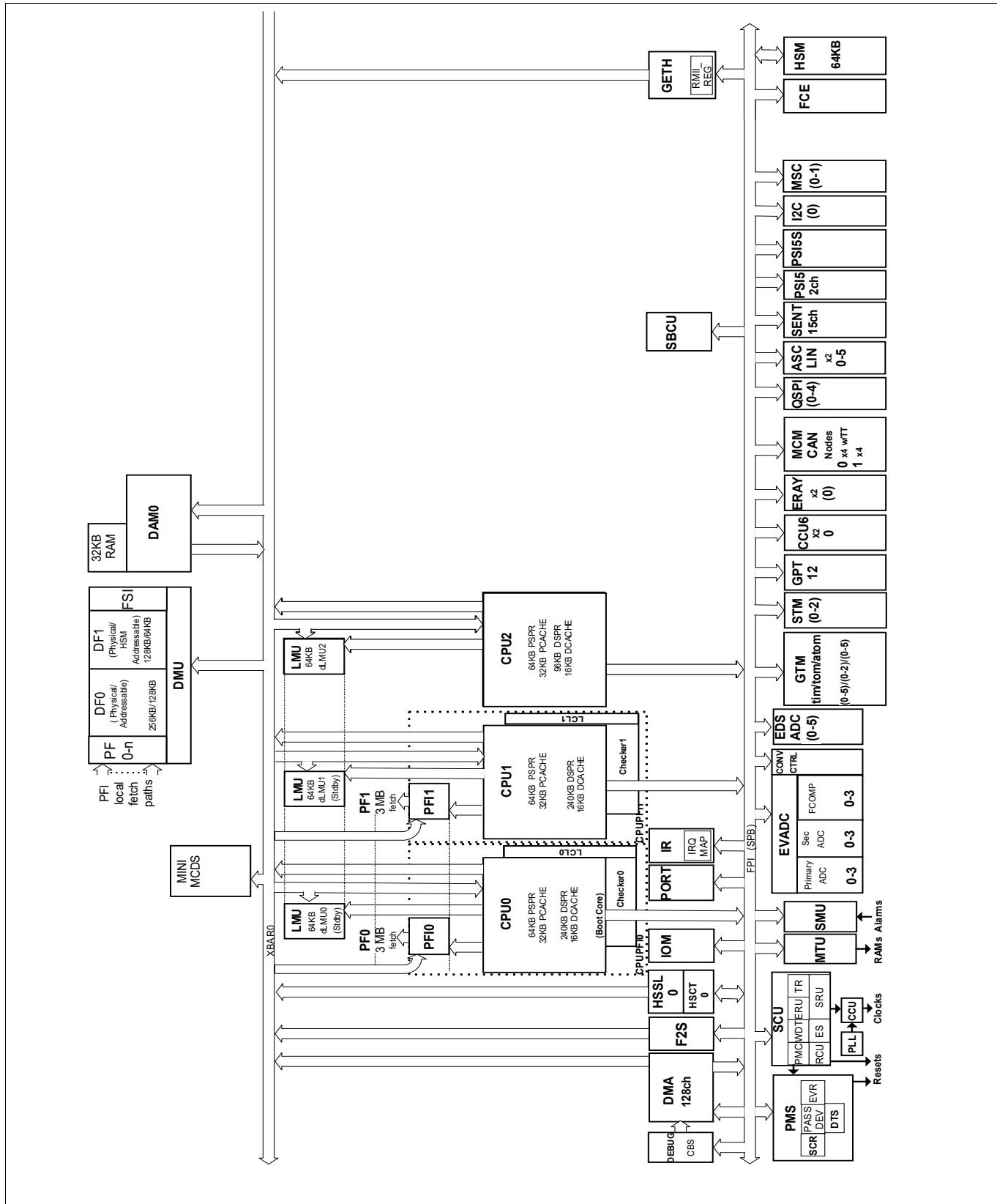
### **1.2.2.2 TC37xEXT Block Diagram**



**Figure 3 Block Diagram of TC37xEXT**

## Introduction

### 1.2.2.3 TC37x Block Diagram



**Figure 4 Block Diagram of TC37x**

## Introduction

### 1.2.2.4 TC36x Block Diagram

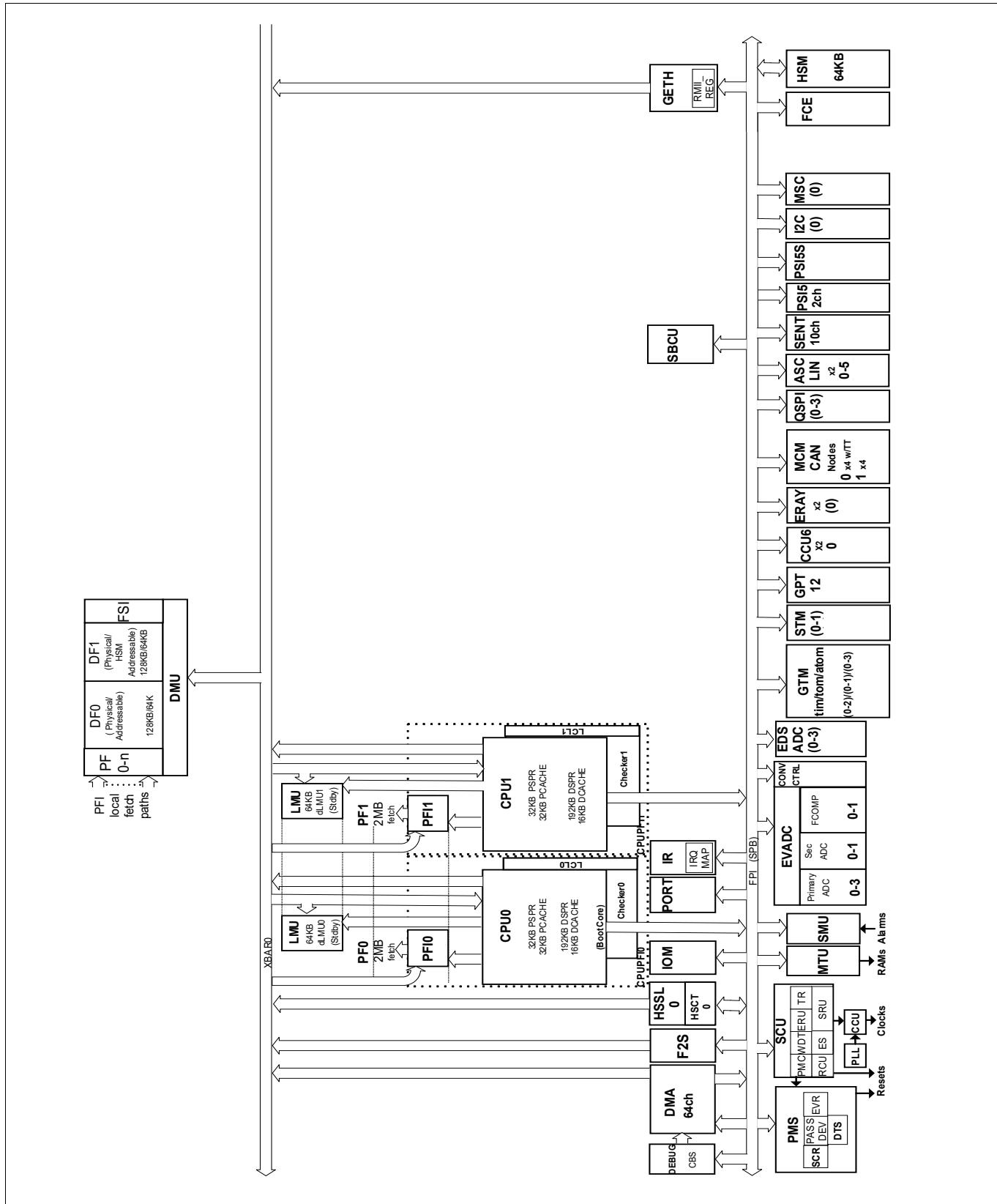


Figure 5 Block Diagram of TC36x

## Introduction

### 1.2.2.5 TC35x Block Diagram

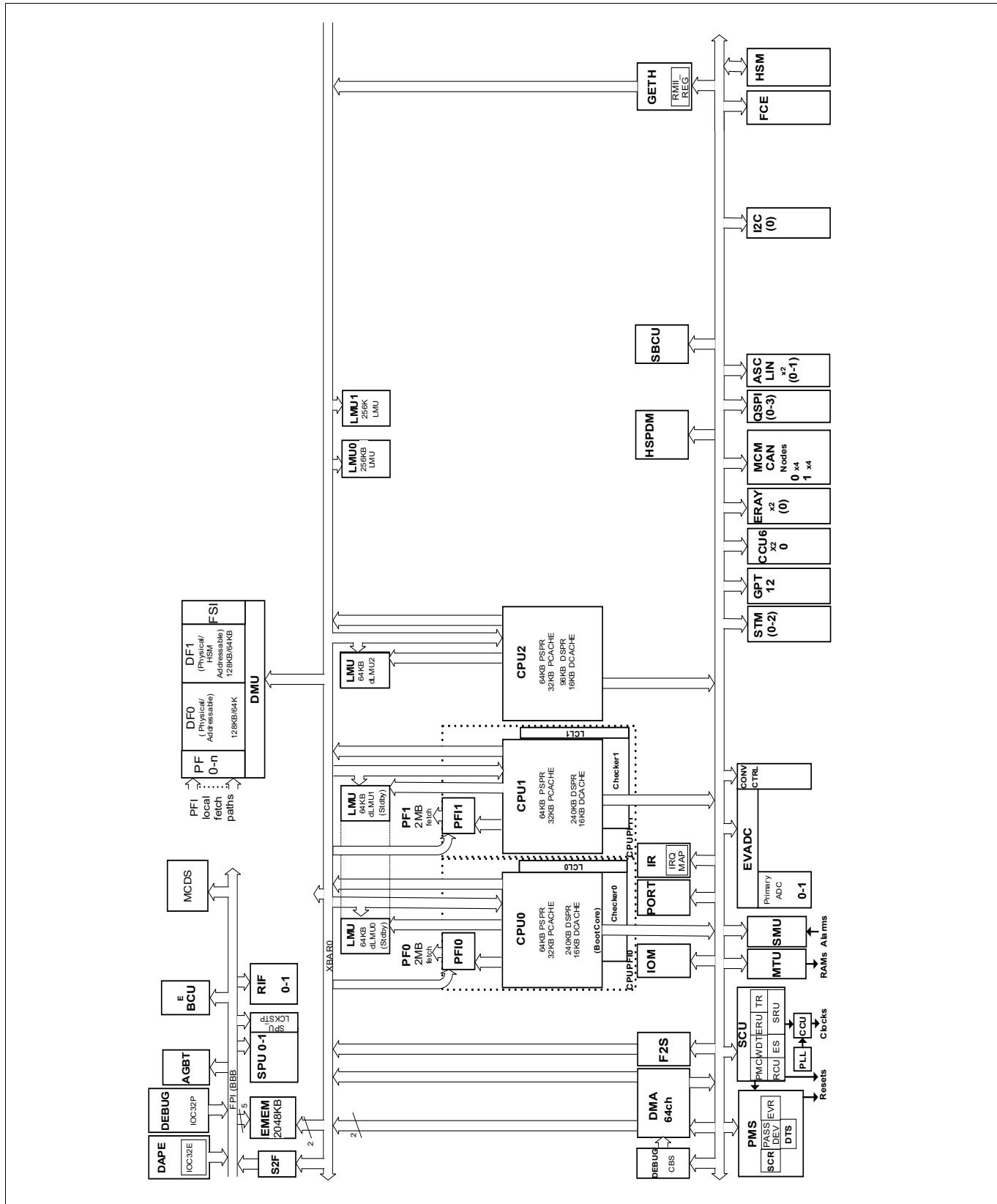


Figure 6 Block Diagram of TC35x

## Introduction

### 1.2.2.6 TC33xEXT Block Diagram

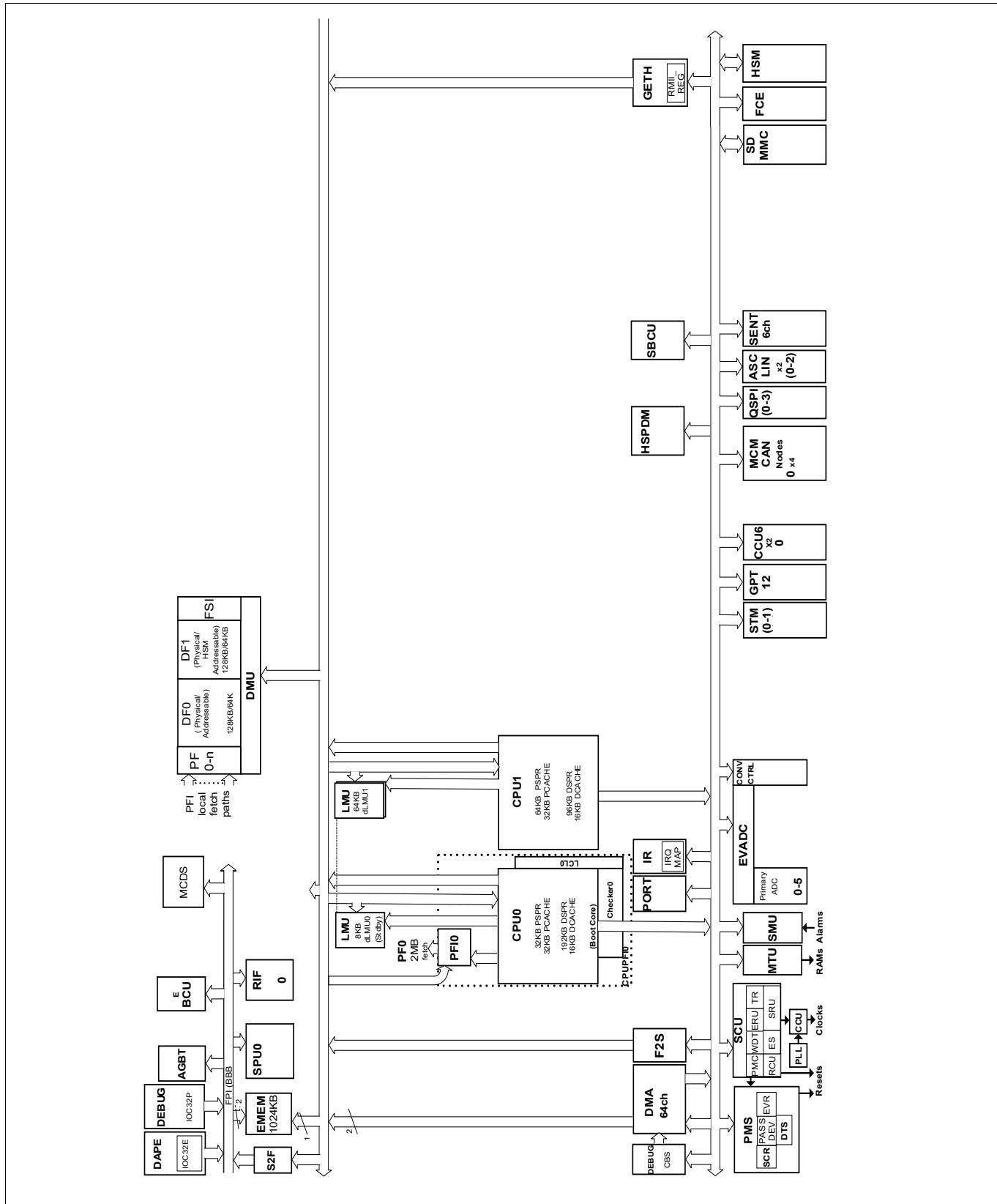
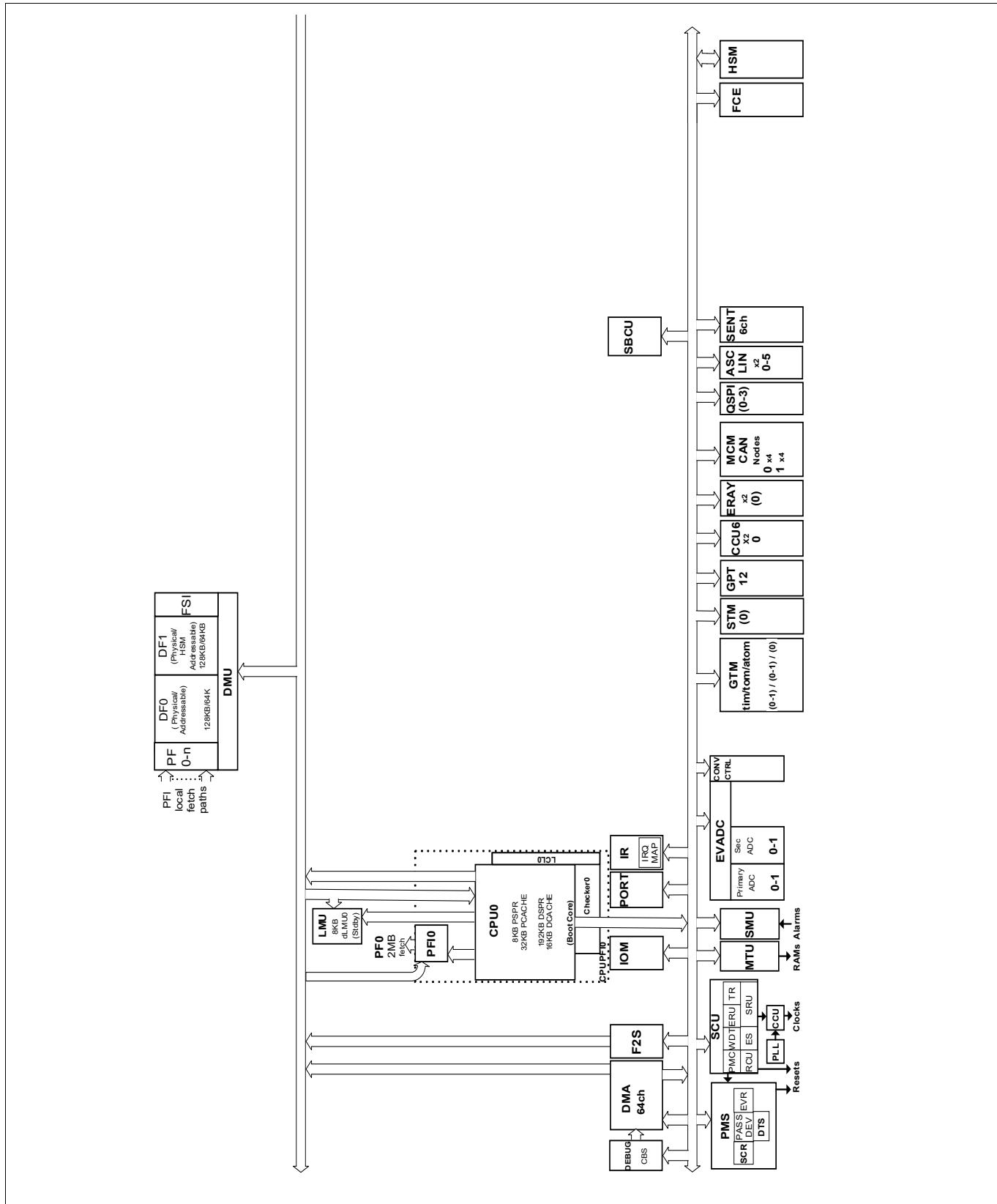


Figure 7 Block Diagram of TC33xEXT

## Introduction

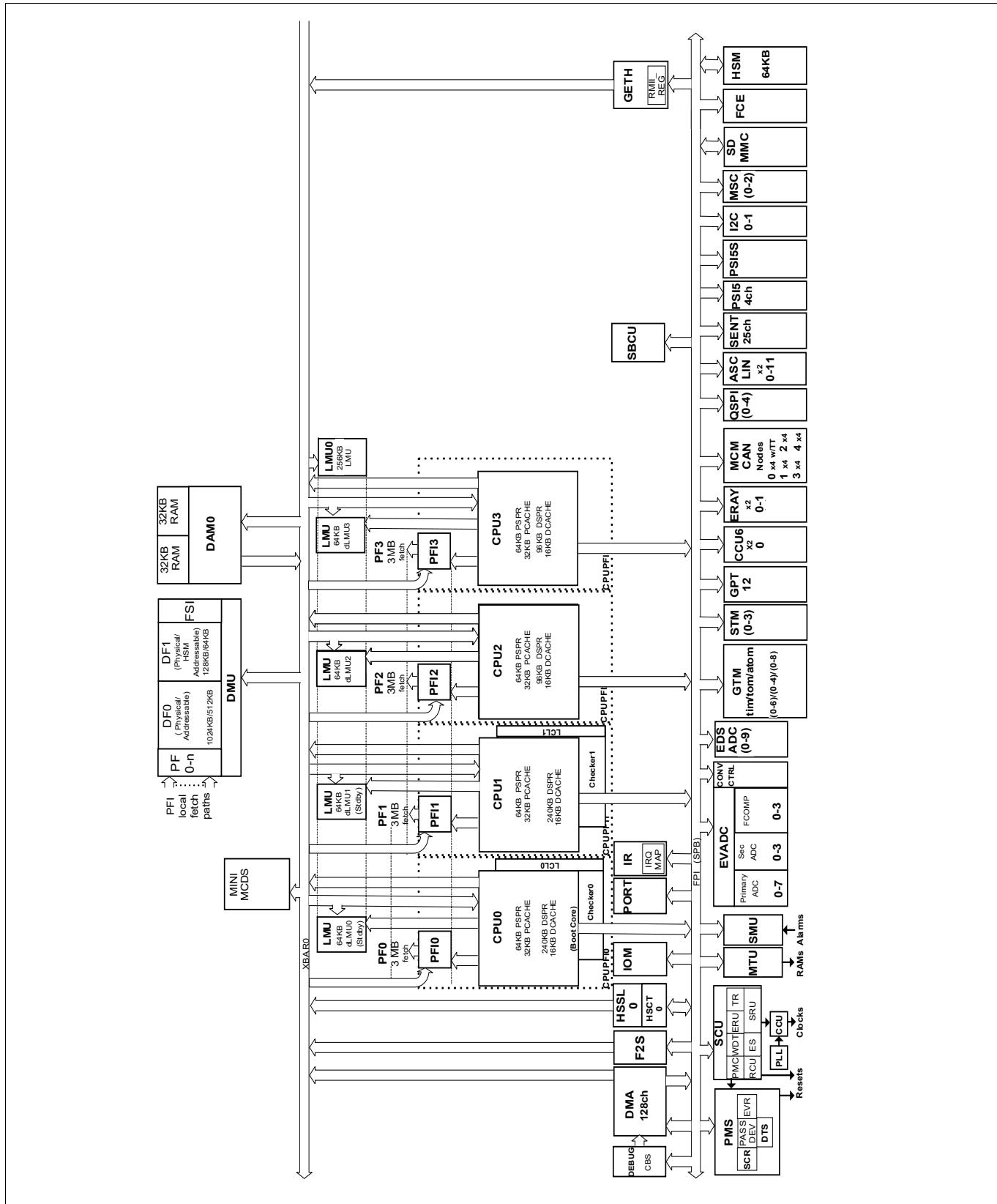
### 1.2.2.7 TC33x Block Diagram



**Figure 8 Block Diagram of TC33x**

## Introduction

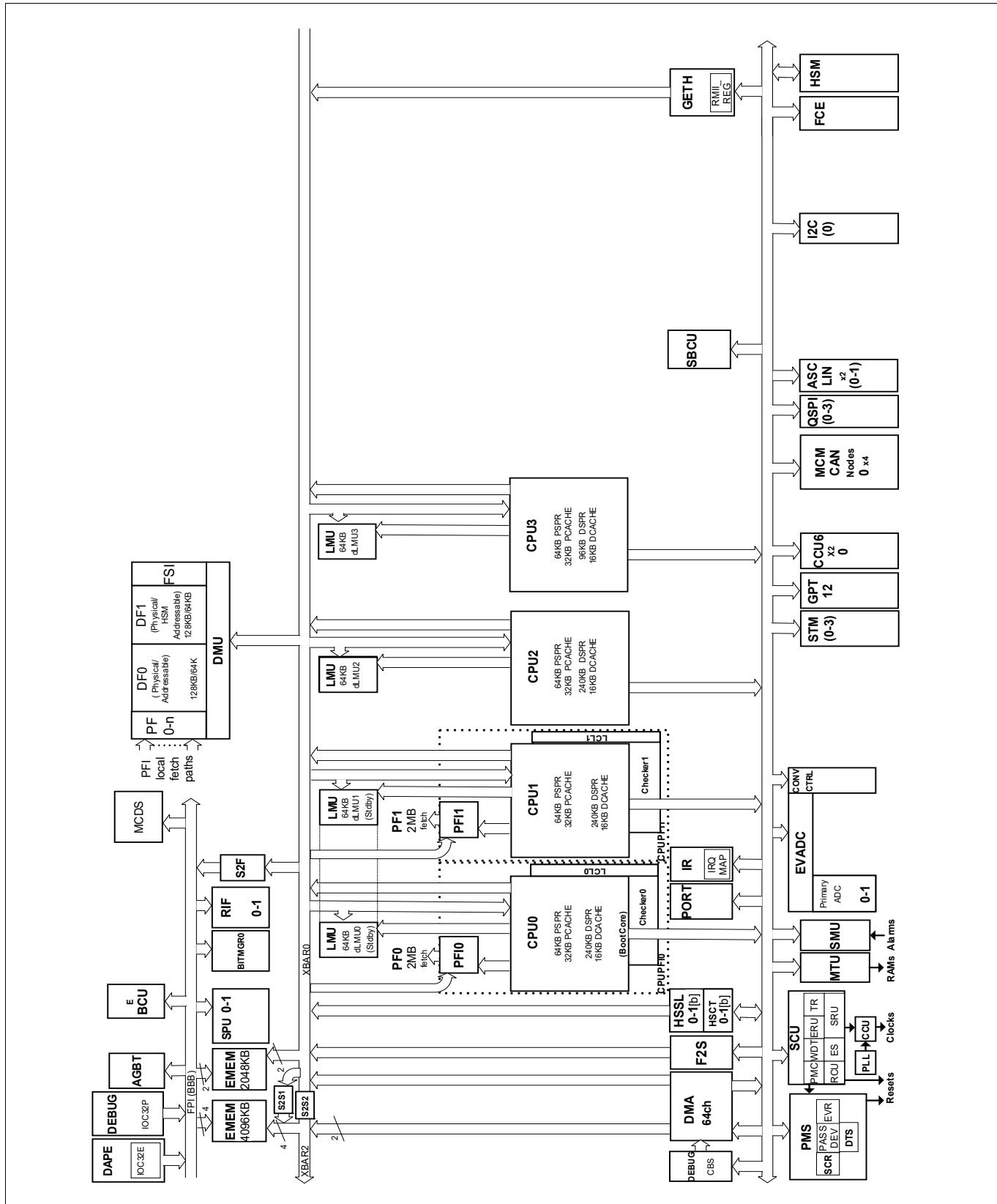
### 1.2.2.8 TC3Ex Block Diagram



**Figure 9 Block Diagram of TC3Ex**

## Introduction

### 1.2.2.9 TC3Ax Block Diagram



**Figure 10 Block Diagram of TC3Ax**

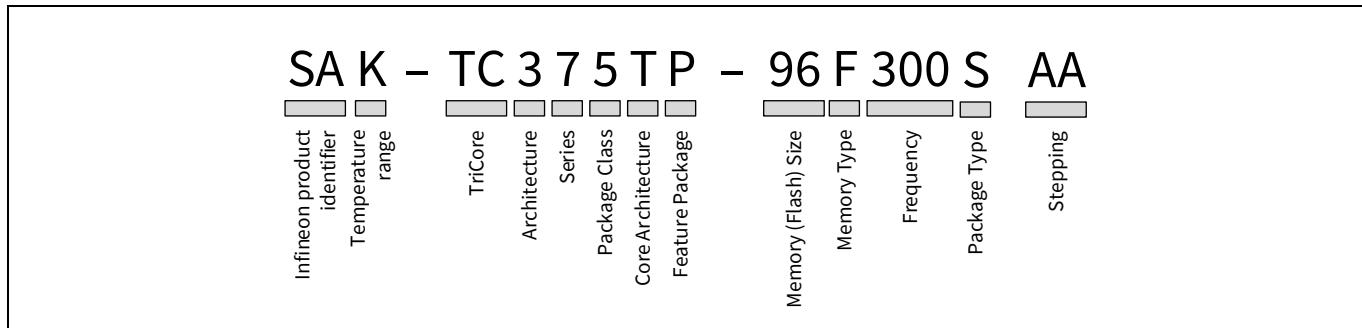
## Introduction

### 1.2.3 Variants

From the devices described in [Chapter 1.2.1](#) and [Chapter 1.2.2](#) “Variants” are created with reduced feature set. This reduction of features is realized by switching functionality off or by documentation only.

#### 1.2.3.1 Encoding of the Product Name

The product name of a variant (e.g. SAK-TC375SB-128F200W AA) supplies the following information:



**Figure 11 Example for Orderable Part Number**

#### Temperature Range

- K :  $T_{\text{ambient}}$  temperature range from  $-40^{\circ}\text{C}$  up to  $+125^{\circ}\text{C}$
- L :  $T_{\text{ambient}}$  temperature range from  $-40^{\circ}\text{C}$  up to  $+150^{\circ}\text{C}$  in packaged devices

#### Architecture

The architecture (= device family) is identified by the “3” in TC3xy.

In AURIX™ TC3xx Platform the SCU\_CHIPID field CHTEC encodes the devices series. Encoding is defined in the User’s Manual.

#### Device Series

The series (e.g. TC39x, TC38x, TC37x, TC35x, TC33x, TC3Ex, TC3Ax) identifies a set of variants.

In AURIX™ TC3xx Platform the SCU\_CHIPID field CHID encodes the device series. Encoding is defined in the User’s Manual.

#### Package Class

The package class is marked with the “x” in the product name, e.g. TC375 with x=”5”. The following classes are defined:

- “9”: 516 pins
- “8”: 233 pins
- “7”: 292 pins
- “6”: 180 pins
- “5”: 176 pins
- “4”: 144 pins
- “3”: 100 pins
- “2”: 80 pins
- “0”: bare die

## Introduction

In AURIX™ TC3xx Platform the SCU\_CHIPID field CHPK encodes the package. Encoding is defined in the User's Manual. It encodes the number of pins and the package type together. Indirectly with a different package also a different pinning is selected however the same package class can have different pinnings, e.g. ADAS and standard.

## Core Architecture

The core architecture identifies the number of TriCore CPUs:

- “X”: Hexa core (= 6 CPUs)
- “Q”: Quad core (= 4 CPUs)
- “T”: Triple core (= 3 CPUs)
- “D”: Dual core (= 2 CPUs)
- “L”: Single core (= 1 CPU)

## Feature Package

The feature package indicates the set of features available in that device. For details the Variants table in the Datasheet Addendum has to be consulted.

Defined feature packages:

- “A”: ADAS extension
  - These products contain the ADAS subsystem, generally with extended memory (EMEM), SPU, RIF. Emulation features are enabled, HSM is enabled when implemented.
- “P”: HSM enabled
  - These products don't contain the ADAS subsystem and not the Emulation features. HSM is enabled.
- “E”: Emulation device
  - These products are emulation devices for the devices of Feature Package “P”. The ADAS peripherals SPU, RIF are not available. HSM is enabled when implemented.
  - Generally Emulation Devices have the same feature set as the to be emulated product device plus additional MCDS functionality.
- “X”: Feature extension
  - These products contain the ADAS subsystem but only with the extended memory (EMEM). The ADAS peripherals SPU, RIF are not available. HSM is enabled when implemented.
- without second letter: HSM disabled, no ADAS extension, no Emulation device, no Feature extension.
  - These products don't contain the ADAS subsystem and not the Emulation features and HSM is disabled.
- The Datasheet Addendum might define additional second letters.

In the SCU\_CHIPID this is encoded. Bit field “EEA” is set in ADAS, “Feature Extension” and Emulation devices. Bit field SEC is set in HSM enabled devices.

## Flash Size Code

The PFlash size is encoded in the product name as multiple of 64 KByte:

- “16”: 1 MByte
- “32”: 2 MBytes
- “48”: 3 MBytes
- “64”: 4 MBytes
- “96”: 6 MBytes
- “144”: 9 MBytes

## Introduction

- “160”: 10 MBytes
- “192”: 12 MBytes
- “256”: 16 MBytes

In the SCU\_CHIPID this is encoded in the FSIZE field.

## Memory Type

The memory type is currently constant “F” for Flash.

## Frequency

The maximum CPU frequency is encoded with this field. Currently defined values are:

- “160” MHz
- “200” MHz
- “300” MHz

## Package Type

The following package types are currently defined:

- “W”: LQFP with 0.5 mm pitch
- “F”: TQFP with 0.4 mm pitch
- “S”: LFBGA with 0.8 mm pitch
- “”: no letter for bare die

In the SCU\_CHIPID this is encoded together with the package class in the field CHPK.

## Stepping

The last two letters identify the stepping, i.e. different versions of the device hardware including ROM changes. In AURIX™ TC3xx Platform the SCU\_CHIPID field CHREV encodes stepping. Encoding is defined in the User’s Manual.

### 1.2.3.2 Emulation Devices

Emulation Devices “ED” (Feature Package “E”) are compatible to the emulated Feature Package “P” and “” (no second letter) devices. “Compatible” means that they offer the same feature set as the Feature Package “P” and “” device with additional debug and tracing capabilities. These devices can contain additional features that are not disabled.

Emulation strategy:

- TC39x Feature Package “E” is based on TC39x device. It can be used to emulate:
  - TC39x Feature Package “P” and “”.
  - TC38x Feature Package “P” and “”.
- TC37x Feature Package “E” is based on TC37xEXT device. It can be used to emulate:
  - TC37x Feature Package “P” and “”.
  - TC36x Feature Package “P” and “”.

### 1.2.3.3 TC32x

The TC32x device series is based on TC33x silicon. The following table shows feature differences to the TC33x which are valid for all TC32x variants. Feature variants of the general TC32x device may be created.

## Introduction

**Table 10 TC32x Feature Overview Delta to TC33x**

Feature	TC32x
CPUs	Max. Freq.
SRAM per CPU	DSPR
Data Flash	DF0 Size (single-ended)
ASCLIN	Modules
	6

### 1.2.3.4 TC39x With Feature Package “P”

The following table shows feature differences between the TC39x silicon and its variants with feature package “P” (see also [Chapter 1.2.3.1](#), section [Feature Package](#)). Reduced feature variants of this general TC39x feature package “P” device may be created.

**Table 11 TC39x Feature Package “P”**

Feature	TC39x Feature Package “P”
Extension Memory (EMEM)	TCM
	XCM
	XTM
SDMMC	eMMC/SD Interface
SPU	Modules
SPU Lockstep Cmp.	Modules
RIF	Modules
HSPDM	Modules
Debug	AGBT

### 1.2.4 Revision History

**Table 12 Revision History**

Reference	Changes to Previous Version	Comment
V1.5.0		
<a href="#">Page 14</a>	Changed for TC33xEXT the number of EVADC converters and channels. Primary converters changed from 4/32 to 6/40 and secondary converters from 2/32 to 0/0.	
<a href="#">Page 15</a>	Corrected for TC33x the GTM configuration of SPE from 0 to 2, of “BRC / DPLL” from “0 / 0” to “1 / 0”, of “CMP / MON” from “0 / 0” to “1 / 1”, of “DTM modules” from “4” to “6 (4 on TOM, 2 on ATOM)”, of “TBU channels” from “2 (TBU0-1)” to “3 (TBU0-2)”.	
<a href="#">Page 15</a>	Removed GTM from TC33xEXT (all GTM features set to 0).	
<a href="#">Page 15</a>	Corrected number of CDTM modules of GTM from 6 to 7 for TC39x. Corrected number of CDTM modules of GTM from 5 to 6 for TC38x and TC38xEXT. Correct the assignment of DTM modules of GTM from “10 on TOM, 10 on ATOM” to “8 on TOM, 12 on ATOM” for TC38x and TC38xEXT.	

## Introduction

**Table 12 Revision History**

Reference	Changes to Previous Version	Comment
<a href="#">Page 15</a>	Removed FlexRay from TC33xEXT.	
<a href="#">Page 16</a>	Changed number of ASCLIN from 12 to 6 for TC33xEXT.	
<a href="#">Page 16</a>	Changed number of CAN modules from 2 to 1 and consequently the number of CAN nodes from “2x4” to “1x4” for TC33xEXT.	
<a href="#">Page 11, 20, 21, 22, 23, 24, 25, 26, 27</a>	Changed in all block diagrams the text in the EVADC sub-blocks. Previous block diagrams showed a non-existing grouping (e.g. “4 x 4”) now the flat converter numbering (e.g. “0-7”) is shown.	
<a href="#">Page 16, 25</a>	Removed IOM from TC33xEXT in Feature Table and its block diagram.	
<a href="#">Page 18</a>	Added support for BGA-180 package with ADAS pinning to TC33xEXT and non-ADAS pinning to TC33x.	
<a href="#">Page 18</a>	Added support for TQFP-80 package with TC33x.	
<a href="#">Page 18</a>	Removed support for TQFP-144 package from TC37xEXT (concrete from TC37x feature package “E” devices). Entered “No” for this package for all TC37x and TC37xEXT devices.	
<a href="#">Page 25</a>	In block diagram of TC33xEXT changed SRI connection of DMA from one interface to two interfaces.	
<a href="#">Page 17</a>	Changed in Feature Table in row “Low Power Feature” the standby RAM capability of TC33xEXT from DLMU0&1 to DLMU0 (same as for TC33x).	
<a href="#">Page 14</a>	Changed in Feature Table in row “DMA” the number of channels from 16 to 64 for TC33x and TC33xEXT.	
<a href="#">Page 16</a>	Changed in Feature Table in row “CAN” the number of modules from 4 to 5 and the numbers of channels from “4 x 4” to “5 x 4” for TC38xEXT.	

### V1.5.1

<a href="#">Page 17</a>	Extended the footnote on AGBT explaining its availability in feature package “E”, “A” and “T” devices.	
<a href="#">Page 15</a>	Changed configuration of GTM of TC33x. Changed from 2 clusters at 100 MHz to 2 clusters at 200 MHz.	
<a href="#">Page 16</a>	Changed documented configuration of SENT of TC39x from “25 + 5 by SW” to “25”. No change of hardware related to this change.	
-	Changed throughout the document the name of the device TC38xEXT to TC3Ex.	
<a href="#">Page 14, 14, 16, 16</a>	Added footnotes to TC3Ex features EVADC, EDSADC, SENT, MSC for which more functionality is implemented than usable in any available package.	

### V1.6.0

<a href="#">Page 12</a>	Added the TC3Ax Feature list to the Table 9.	
<a href="#">Page 28</a>	Added the TC3Ax Block Diagram.	
<a href="#">Page 29</a>	Added TC3Ax to the Device Series list.	
<a href="#">Page 29</a>	Added “80 pins” and “233 pins” in “Package class” options.	

## Introduction

**Table 12 Revision History**

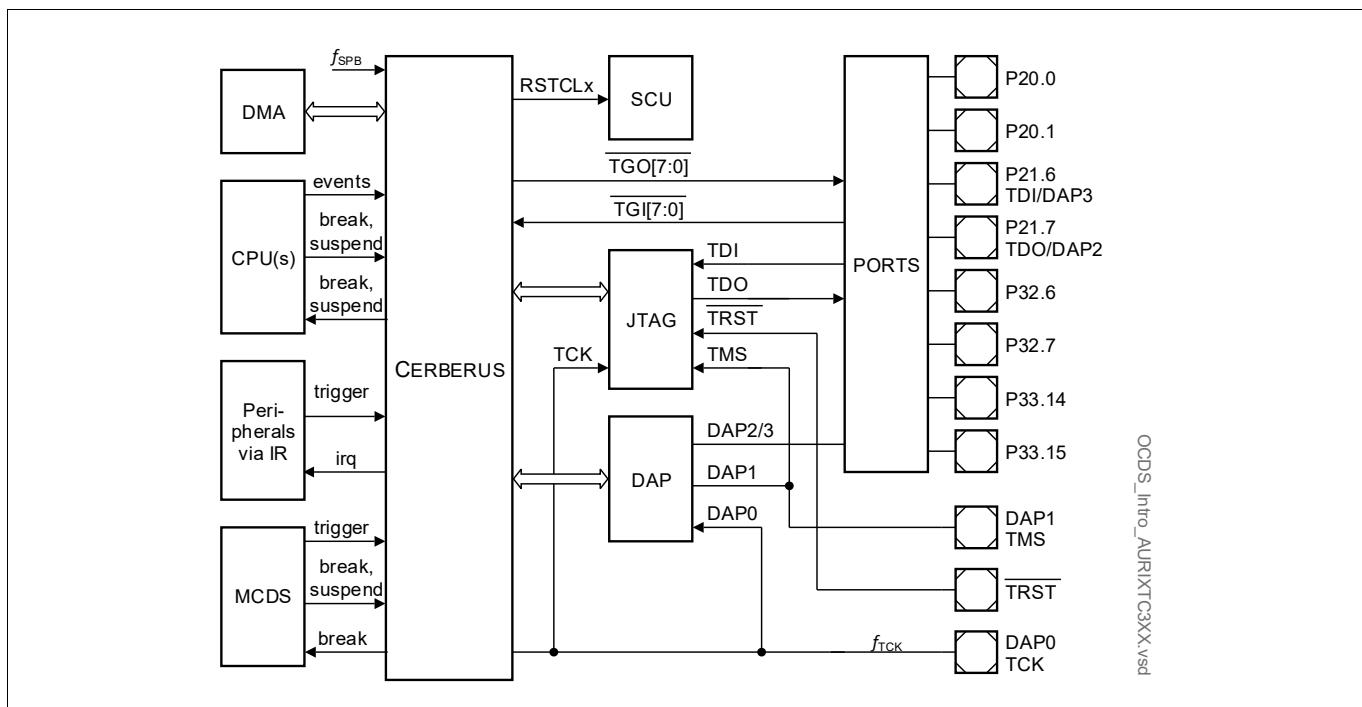
Reference	Changes to Previous Version	Comment
<b>Page 12</b>	Added the row “Power Management System” to the Platform Feature Overview- Table 9.	
	FlexRay for the TC3Ax is not available. Updated Platform Feature Overview- Table 9.	
<b>Page 2</b>	Table 1 ‘Note’ description updated for DAM RAM.	
<b>Page 1</b>	Update kbit, Mbit definitions to kbit/s and Mbit/s respectively in the “Text Conventions” section.	
<b>Page 28, Page 12</b>	Update the TC3Ax Block Diagram and the Feature List Table with the BITMGR0.	
<b>V1.6.1</b>		
<b>Page 31</b>	In chapter 1.2.3.3 removed the wording ‘Reduced’ and replaced with ‘Feature variants of the general TC32x device may be created’.	
<b>Page 28</b>	Updated the TC3Ax Block Diagram with the XBAR2 and CPU2 DSPI size.	
<b>Page 12</b>	Updated the CPU2 DSPI Size for TC3Ax in “Platform Feature Overview - Table 9.	
<b>V1.6.2</b>		
<b>Page 12</b>	Updated Platform Feature Overview- Table 9. with TC33x EVADC Secondary Groups / Channels to 2 /28.	
<b>Page 29</b>	Definition of "Feature Package" - "X" is updated.	
<b>Page 12</b>	Updated the Feature ‘Packages’ to include BGA216 package support for TC3Ax in “Platform Feature Overview” - Table 9.	

## 1.3 On-Chip Debug Support (OCDS)

The Infineon AURIX™ TC3XX devices contain powerful resources for debugging and performance optimization. They provide high visibility and controllability of software, hardware and system, especially also under hard real-time constraints. The resources are either embedded in specific modules (e.g. breakpoint logic of CPUs) or part of the central Cerberus module.

**Figure 12** shows the AURIX™ TC3XX family concept of the OCDS with all potentially usable pins. The same OCDS function is always on the same port pin including full DAP/JTAG functionality but the number of TGI/TGO trigger pins depends on the device type.

Trace functionality is available for TC3XX with miniMCDS, MCDSlight or MCDS. Please refer to the table “Platform Feature Overview” for more details.



**Figure 12 OCDS Block Diagram (Family Concept)**

When acting as a bus master Cerberus shares the master interfaces of the DMA (using a different master tag) to access memories and SFRs attached to the SPB or SRI via the shortest possible path.

### 1.3.1 Introduction

#### Debugging

The application is not yet ready, the system outside of the TC3XX is either not connected or under control by other means so that misbehavior of the software has no catastrophic consequences. The “user” is a software design engineer with thorough knowledge of the device and the system, in other words, no protection is needed. From the tool the user expects

- Download: The memories of the SoC (and of other external memories attached to the SoC) must be written (and programmed in case of non-volatile memories) without need to disassemble the application system.
- Run Control: Each processor core can be stopped and started at will, either separately or synchronously throughout the SoC.

- Visibility: The content of all storage locations inside the SoC, i.e. memories, SFRs and processor registers, can be read and written, preferably even while the system is running.
- Trace: A log of the processing is desired, as detailed (“cycle accurate”) and wide (aligned trace of parallel processes) as possible.

**Note:** *Due to the high speed of the TC3XX tracing has limitations in trace buffer with the standard chip, but a pin and package compatible Device with extended trace capabilities is available.*

### Rapid Prototyping

The application design has reached a state where operation at target speed is possible. The SoC’s tasks are mission critical, but some of the algorithms are still in development. Hardware support is needed for

- Triggering: An external high speed processor is attached to the SoC to perform all processing tasks not yet implemented inside the SoC. This external system must be notified (“triggered”) whenever its services are needed, possibly with detailed information on the kind of service requested.
- Data Exchange: When triggered, the external “bypass” processor must get hold of the input data of its algorithm as fast as possible. After the calculation the results must be written back into the SoC, again with low latency and possibly triggered by another event. Other tasks of the SoC shall not be influenced.

### Calibration

Once the algorithms are fixed there still is need to “tweak” the software. Namely the constants used by the program typically are dependent on external parameters and are “tuned” in the final application, i.e. in the field or even a driving car.

- Overlay: As the constants are stored in non-volatile memory changing them requires an erase-program cycle, which is only possible safely when the application is taken out of service. Therefore RAM is mapped into the address space “over” locations the software addresses as ROM. This RAM can be accessed and changed by the tool concurrently.

**Note:** *The additional RAM offered by the TC3xxED or TC35x, TC3Ax Devices can be used to extend the amount of overlaid ROM considerably.*

- Measurement: To find points still needing attention or simply to judge the results of parameter changes significant internal data of the SoC (variables, sensor data) must be read from the SoC with deterministic timing and high bandwidth.

**Note:** *The MCDS respectively MCDSlight module of the TC3xxED or TC35x, TC3Ax Devices can be used to capture relevant data highly selective and without modification of the application software.*

### 1.3.2 Feature List

Please also refer to debug specific features in other modules, e.g.

- Eight hardware breakpoints for TriCore, based on instruction or data address
- Unlimited number of software breakpoints (DEBUG instruction)
- Trigger generated by the access to a specific bus address by any bus master
- Peripheral trigger and trace with OCDS Trigger Bus (OTGB)
  - Peripherals provide vectors (Trigger Sets) of their most interesting signals.
  - Signals can be routed to trigger pins.
  - Flexible tracing of signal vectors from one to three peripherals in parallel

- Dedicated interrupt resources to handle debug events, both local inside the CPUs (breakpoint trap, software interrupt) and global (triggered by Cerberus), e.g. for implementing monitor programs

### Central functions implemented by Cerberus

- Run/stop and single-step execution independently for each CPU
- Run/stop and time-step execution of the complete device using the Trigger Switch
- Automatic suspension of CPU associated watchdogs and system timers if the CPU is halted by the tool
- All kinds of reset can be requested using only the tool interface.
- Halt-after-Reset for repeatable debug sessions
- Tool access to all SFRs and internal memories independent of the CPUs
- Bus priority of Cerberus can be chosen dynamically to minimize real-time impact.
- Up to 8 package pins can be used optional as with Trigger In/Out ( $\overline{\text{TGI}}/\overline{\text{TGO}}$ ).
- Central OCDS Trigger Switch (OTGS) with 7 independent Trigger Lines to collect debug events from various sources (all CPUs, DMA, all interrupt requesters, bus controllers, several complex peripherals, MCDS, trigger input pins) and distribute them selectively to all CPUs, DMA and trigger output pins
- Central Suspend Switch using up to three Lines of the Trigger Switch infrastructure. This allows to selectively suspend all or only part of the CPUs and peripherals instead of halting them as reaction to any debug event.
- Access to all OCDS resources also for the CPUs themselves for debug tools integrated into the application code.
- Triggered Transfer of data for simple variable tracing
- A dedicated trigger bank (TRIG) with 96 independent status bits is provided to post requests at a central location from application code to the tool.
- The tool is notified automatically when the trigger bank is updated by any processor. No polling via a system bus is required.
- Fault and stress injection for testing the robustness of a system

### Tool Interfaces

Several options exist for the communication channel between tools and devices:

- DAP and JTAG are clocked by the tool.
- Two pin DAP (Device Access Port) protocol for long connections or noisy environments
- Three pin DAP Unidirectional Mode for off-chip transceiver integration (e.g. LVDS)
- Three pin DAP Wide Mode for high bandwidth needs
- Four pin DAP Unidirectional Wide Mode for off-chip transceiver integration with high bandwidth needs
- DAP bit clock can have any frequency up to 160 MHz.
- 15 MByte/s for block read or write, 25-30 MByte/s in Wide Mode and Unidirectional Wide Mode
- Optimized random memory accesses (read word within 0.5  $\mu$ s at 160 MHz)
- CAN (plus software linked into the application code) for embedded purposes with lower bandwidth requirements
- DAPE can be used in parallel to DAP to connect a second tool for Emulation Devices.
- Four pin JTAG (IEEE 1149.1) for standard manufacturing tests
- Lock mechanism to prevent unauthorized tool access to application code

- Hot attach (i.e. physical disconnect/reconnect of the host connection without reset) for all interfaces
- Infineon standard DAS (Device Access Server) implementation for seamless, transparent and parallel tool access over any supported interface
- DAP over CAN Messages (DXCM)

## FAR Support

To efficiently locate and identify faults after integration of an AURIX™ TC3XX device into a system special functions are available:

- Boundary Scan (IEEE 1149.1) via JTAG or DAP.
- SSCM (Single Scan Chain Mode) for structural scan testing of the chip itself.
- DXCPL (DAP over CAN Physical Layer) via CAN pins (AP32264)

*Note: Boundary scan is possible also for locked devices. The security barrier is within CERBERUS.*

### 1.3.3 Family Overview

The OCDS architecture and features are very consistent over the whole AURIX™ TC3XX family. This makes it easy for tool partners and users to switch between different devices. The following tables list all JTAG IDs.

**Table 13 JTAG IDs of AURIX TC3xx Devices**

	<b>TC39x</b>	<b>TC38x</b>	<b>TC37x</b>	<b>TC37xEXT</b>	<b>TC36x</b>
A-step	10205083 <sub>H</sub>	10206083 <sub>H</sub>	10207083 <sub>H</sub>	10208083 <sub>H</sub>	10209083 <sub>H</sub>
B-step	20205083 <sub>H</sub>				

**Table 14 JTAG IDs of AURIX TC3xx Devices (continued)**

	<b>TC35x</b>	<b>TC33xEXT</b>	<b>TC33x</b>	<b>TC3Ex</b>	<b>TC3Ax</b>
A-step	1020A083 <sub>H</sub>	1020C083 <sub>H</sub>	1020B083 <sub>H</sub>	10215083 <sub>H</sub>	1020D083 <sub>H</sub>

### 1.3.4 Tool Interface Recommendations

AURIX™ TC3XX devices are well supported by many tool partners for different types of tools. Standard tool interface for debug, measurement and calibration is DAP due to its reduced pin-count, higher performance (3-6x) and higher robustness (CRC) than JTAG. Please note that the full “JTAG” boundary scan functionality is also available with DAP, it is supported however only by specific tool providers. Please refer to application note AP24003 for more information about the standard DAP connector and board design for high speed DAP. [Table 15](#) lists all pins and considerations for connecting tools.

**Table 15 Tool Relevant Device Pins of AURIX™ TC3XX Family**

<b>Pins</b>	<b>Remark</b>
TRST/DAPE0	DAP: Has to be high at PORST pin release. TRST has a pull-up. JTAG: Needs to be controlled by the tool via the tool connector. The DAPE interfaces can be operated in parallel with the DAP for Emulation Devices.
DAP0/TCK	DAP: Please consider AP24003 for high speed DAP
DAP1/TMS	
DAP2/TDO/DAPE2 TGI3/TGO3	DAP: Needed for three pin modes like high bandwidth Wide Mode and Unidirectional Wide Mode. The standard DAP connector (AP24003) allows to use this pin on demand either for Wide Mode e.g. for measurement or as trigger pin for system debugging. The DAPE interfaces can be operated in parallel with the DAP for Emulation Devices.
DAP3/TDI/DAPE1 TGI2/TGO2	DAP: Needed for four pin mode like high bandwidth Unidirectional Wide Mode. The DAPE interfaces can be operated in parallel with the DAP for Emulation Devices.
VDDSB	1.25 V supply of the ED memory. <ul style="list-style-type: none"><li>• VDDSB has to be supplied when VDD is supplied and the EMEM is unlocked.</li><li>• VDDSB can be unsupplied when VDD is supplied and PORST is active or the EMEM is locked.</li><li>• VDDSB can be supplied when VDD is unsupplied and PORST is active (EMEM standby mode).</li></ul>
VEXT	3.3 V or 5 V external power supply for the device. VEXT decides also the DAP connection voltage.

**Table 15 Tool Relevant Device Pins of AURIX™ TC3XX Family (cont'd)**

Pins	Remark
TGlx/TGOx	Optional trigger pins, overlaid to port pins. Availability depends on device and package type.
AGBT_xyz	AGBT high-speed serial pins in the center ball matrix of EDs in BGA packages. Please connect to VSS if no AGBT is needed.

*Note:* Dedicated DAPE pins are provided for Emulation Device packages, referring to the data sheet and the ED specification.

For more information please contact your Infineon support.

### 1.3.5 Debug Access Server (DAS)

The DAS API provides an abstraction of the physical device interface for tool access. The key paradigm of DAS is to read or write data in one or several address spaces of the target device.

#### DAS Features

- Standard interface for all types of tools
- Efficient and robust methods for data transfer
- Standardized system security support (authorization)
- Several independent tools can share the same physical interface
- Product chip address space is represented with DAS address map 0, EEC with 1
- Infineon's miniWiggler supports DAP, JTAG, SWD and SPD

DAS is not device specific. It can be used for all Infineon 8-, 16- and 32-bit microcontrollers with DAP, JTAG, SWD, or SPD interface. For more information please refer to [www.infineon.com/DAS](http://www.infineon.com/DAS).

### 1.3.6 Revision History

**Table 16 Revision History**

Reference	Change to Previous Version	Change Request Comment
<b>V3.1.11</b>		
	This the first release.	
<b>V3.1.12</b>		
<a href="#">Page 36</a>	Term "Emulation Device" corrected in several notes (only editorial changes).	
<a href="#">Page 39</a>	Table <a href="#">JTAG IDs of AURIX TC3xx Devices</a> updated.	
<b>V3.1.13</b>		
<a href="#">Page 39</a>	Name of the device TC38xEXT changed to TC3Ex.	0000056760-67
<b>V3.1.14</b>		
<a href="#">Page 38</a>	Note about boundary scan for locked devices added.	0000056761-35
<a href="#">Page 39</a>	Name of the device TC35xEXT changed to TC3Ax.	0000056760-67
<a href="#">Page 36</a>	TC3Ax added to Notes in Calibration description	0000056760-67

## 1.4 Emulation Device (ED)

In the AURIX TC2xx family for most of the so-called production devices a corresponding pin-compatible TC2xxED Emulation device is available. An Emulation Device comprises the unchanged Product Chip Part (SoC) and the Emulation Extension Chip (EEC) part ([Figure 13](#)).

In the AURIX®2G TC3xx family this is similar from a functional perspective. The EEC part contains as well an Extension Memory (EMEM) and a Trace module. The MCDS trace module is on a TC39xED (device name TC39x) and TC37xED (device name TC37xEXT) while the MCDSlight trace module is on a TC33xED (device name TC33xEXT) and a TC35x. Devices with ED functionality are devices with “Feature Package E”.

**Table 17 TC39x/37x/33xED and TC35x, TC3Ax Comparison**

Functionality / Device name	TC39xED / TC39x	TC37xED / TC37xEXT	TC33xED / TC33xEXT	TC35x, TC3Ax
Emulation device for	TC39x TC38x TC3Ex	TC37x TC36x TC33x	TC33xED	TC35x, TC3Ax
MCDS variant	MCDS	MCDS	MCDSlight	MCDSlight
EMEM RAM total (incl. XTM)	4112 KB	3088 KB	1040 KB	2064 KB
EMEM TCM	2048 KB	2048 KB	1024 KB	2048 KB
Packages	Please refer to the Platform Feature Overview table in the TC3xx family documentation for the latest information about available packages	Please refer to the Platform Feature Overview table in the TC3xx family documentation for the latest information about available packages	Please refer to the Platform Feature Overview table in the TC3xx family documentation for the latest information about available packages	Please refer to the Platform Feature Overview table in the TC3xx family documentation for the latest information about available packages
AGBT (Aurora)	yes	yes	yes	yes

As shown in [Table 17](#) the TC39xED is the Emulation device for TC38x and TC3Ex. Thermal design and power supply need to accomodate the higher TC39xED power consumption i.e. higher leakage current due to a larger chip (in particular for the EMEM) and higher dynamic current if additional resources (e.g. MCDS) are being used.

### Requirements for TC38x code:

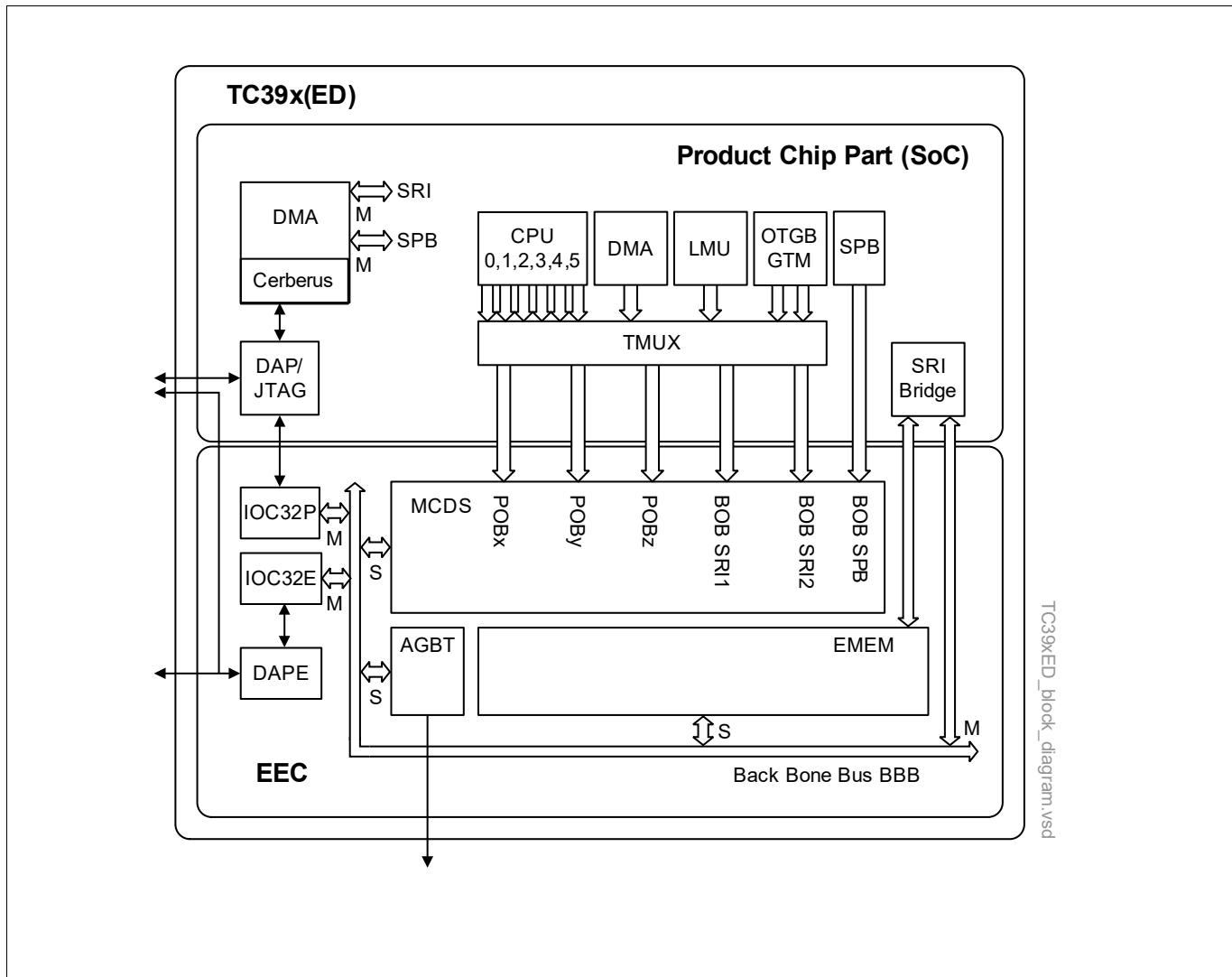
- It needs to be tolerant to the different CHIP and JTAG ID values.
- It may not access reserved address ranges and expect an error if these ranges are present for TC39xED.
- It may not access the miniMCDS subsystem.
- It may not access the upper 12 ASCLIN modules (out of 24).

### Additional requirements for TC3Ex code:

- It may not access the upper 2 out of the 5 CAN modules.
- A workaround is needed if the external clock source pin is used for the RTC in the SCR.

### 1.4.1 Block Diagram

**Figure 13** shows the block diagram of TC39xED as an example. The Product Chip Part is reduced to the directly connected modules only in the drawing. Please refer to **Table 18** for an explanation of the acronyms.



**Figure 13 Block Diagram**

**Table 18 TC39xED Components (Figure 13)**

<b>Component</b>	<b>Definition</b>
AGBT	Aurora GigaBit Trace module on EEC
BBB	Back Bone Bus with FPI protocol
BOB	Bus Observation Block. Trace and trigger logic within MCDS
Cerberus	Central debug and tool access control unit is part of OCDS
DAP	Device Access Port. Fast and robust 2/3 pin tool interface (15/30 MB/s).
DAPE	Independent DAP instance for connecting a second tool to the ED
DMA	Direct Memory Access controller. Shares SRI/SPB bus interfaces with Cerberus.
ED	Emulation Device for calibration, measurement and debug
EEC	Emulation Extension Chip part
EMEM	Extension Memory (calibration and trace memory)
FPI	Flexible Peripheral Interconnect, the protocol of SPB and BBB buses
IOC32P/E	IO Client for accessing BBB by DAP/DAPE
MCDS	Multi-Core Debug Solution
OCDS	On-Chip Debug Support
OTGB	OCDS Trigger Bus: collects interrupt and peripheral trace and trigger signals.
POB	Processor Observation Block
SBCU	SPB Bus Control Unit
SoC	System on Chip (used also for “product chip part”)
SPB	System Peripheral Bus with FPI protocol
SRI	Shared Resource Interconnect cross bar
TMUX	Trace Multiplexer

The Extension Memory (EMEM) as part of the EEC is used for two conceptually different purposes: trace buffering and overlay memory. The size allocation for both parts can be configured in the EMEM module.

For calibration, RAM partitions are mapped into the address ranges of the CPUs, optionally replacing parts of the TC3xx's local Flash. The feature is described in detail in the Overlay section of the TC3xx manual.

Tracing on the other hand is a non intrusive tool to aid the debugging process. Matching elements from the MCDS module are provided to translate the captured signals - routed there from CPUs and other sources - into trace messages. These messages are buffered in EMEM and can then be read by the tool: e.g., via DAP.

### 1.4.2 Feature List

This section lists the features for the TC3xxED and TC35x, TC3Ax devices.

#### Applications

- Software development
  - Debugging
  - Performance analysis and optimization
- Calibration
- Measurement
- Rapid prototyping

#### General Features

- The behavior of a TC3xx device (e.g. TC37x or TC36x) and its corresponding TC3xxED device (e.g. TC37xED) is identical with restrictions listed below **Table 17**
- The package of a TC3xx device and its corresponding TC3xxED device is footprint compatible device
- Minimum number of ED specific pins
- Full access to the EEC part via the regular DAP/JTAG package pins
- No external emulator hardware required other than DAP/JTAG interface
- Tool software running on TriCore has full EEC access e.g. for calibration or measurement
- Protection against reverse engineering by competitors and against manipulation, both for production and emulation device (field trials)
- Additional DAP interface (DAPE) for connecting a second independent tool
- High speed Aurora GigaBit Trace (AGBT) interface

#### Extension Memory Features

- Extension Memory (EMEM) size is up to 4112 KB
- EMEM has two 8 KB XTM tiles, up to eight 256 KB TCM tiles, and up to 2 MB XCM
- EMEM except XTM can be used for calibration, code, constants, or data storage
- Up to 2048 KB can be used for trace buffering
- Continuous trace via DAP / AGBT requires just two XTM tiles (only one XTM tile for AGBT on TC39xA-step)
- Support of independently operating calibration and debug tools
- FIFO functionality for continuous trace (EMEM address triggers in MCDS)
- EMEM is also mapped into the address range of TriCore
- EMEM can be overlaid to Flash
- Code and data fetch from EMEM
- Data retention of RAM during power down by isolated standby power supply
- ECC with SECDED (Single Error Correction, Double Error Detection)

#### Measurement Features

- Highly efficient SW triggering via DAP interface (TRIG)
- Fine Grained Trace Qualification for low-cost Trace Based Measurement (TBM) via DAP
- Aurora GigaBit Trace (AGBT) interface for high-end trace based measurement

## Debug Features

- TriCore program trace (instructions, program flow, functions only)
- Continuous Compact Function Trace (CFT) via DAP
- TriCore data trace (no register file trace)
- Parallel trace of three CPUs, two selected SRI clients, and the SPB bus
- Full visibility of internal peripheral bus (SPB)
- Time aligned trace of all sources
- Trace of internal states and signals of complex peripherals
- Trace of interrupt and DMA requests and processing
- Trace of EVR state and control loop signals
- Breakpoints and watch points based on common event generation logic
- Magnitude comparators working on instruction pointers and memory addresses: A <= IP <= B
- Masked magnitude comparators working on the data busses: DATA = "xxxx55xx"
- Sequential event logic: counters driven by events and equipped with limit comparators are used as event sources again for breakpoint or trace qualification
- Optimized compression of buffered trace data
- Very powerful qualification- and trigger mechanism
- Pre- and post-event trace buffering ("digital oscilloscope")
- Performance counters
- Concurrent trace logging and trace data acquisition up to the bandwidth of the used host interface
- Central time stamp unit to correlate traces from different CPUs and other sources
- Halt the system or parts of it when trace memory is full
- Regular and modular structure of the control blocks and registers
- Trace debug unit power reduction modes with clock gating
- Trace data in EMEM can be decoded after unsolicited PORST
- Output of continuous trace over DAP/DAPE or Aurora GigaBit Trace interface
- HSM bus traffic completely filtered away by default

### 1.4.3 Comparison to AURIX Emulation Devices

TC3xx Emulation Devices are designed as close to the predecessors of the AURIX family as feasible.

#### New Features

- CPU read data value trace
- Trace based measurement (TBM) mode
- Second DAP interface (DAPE) for connecting two tools in parallel
- Unidirectional Wide Mode for DAP interface only (not for DAPE)
- Trace of EVR state and control loop signals
- Option to make HSM accesses visible in SPB trace
- Debug support of EBCU enabled
- EMEM standby supply status bit (SBRCTR.STBPON)

#### Changes

- DAP/DAPE pins supplied by regular VEXT voltage (5V / 3.3V)
- DAP/DAPE pins use TTL level
- Center ball matrix of BGA packages includes dedicated DAPE pins
- Program trace of up to three CPUs in parallel
- Local trace source multiplexer in CPU subsystem (TCMUX)
- Data trace of up to five CPUs (or four + DMA) in parallel for Trace Based Measurement
- GTM has two additional direct 32 bit trace busses to MCDS (OTGBM0/1)
- Improved trace data streaming via DAP (NTN NOW capture)
- BBB access with an SRI to FPI bridge from CPUs (not via LMU)
- EMEM access with SRI to SRI bridge from CPUs (not via LMU)
- EMEM size is up to 4 MB
- EMEM access latency reduced for CPUs
- Up to 2 MB EMEM can be used for trace buffering
- EMEM TCM tile size is 256 KB
- EMEM XTM addressing changed due to changed TCM tile size
- EMEM XTM can only be used for trace data (MCDS, BBB, AGBT). No CPU access anymore.
- EEC addresses are now the same from IOC32P/E and from CPU point of view
- Prolog Code PCEDS addresses changed

#### Discontinued

- Cold start EMEM access via Standby DAP
- FusionQuad™ packages

## 1.4.4 Trace Source Multiplexer

**Figure 13** shows the TMUX multiplexer between the different trace sources and the five flexible observation blocks of the MCDS (BOB SPB is dedicated for SPB). Within the CPU subsystem there is another local multiplexer (TCMUX).

### 1.4.4.1 TMUX Setting Options

For keeping the on-chip wiring of wide high-speed trace busses within reasonable bounds, the trace source setting options for some MCDS observation blocks are significantly reduced (**Table 19**). This reduced set was selected taking into account the use cases for parallel trace listed in [Section 1.4.4.3](#).

**Table 19** Trace Source Multiplexer (TMUX) Setting Options

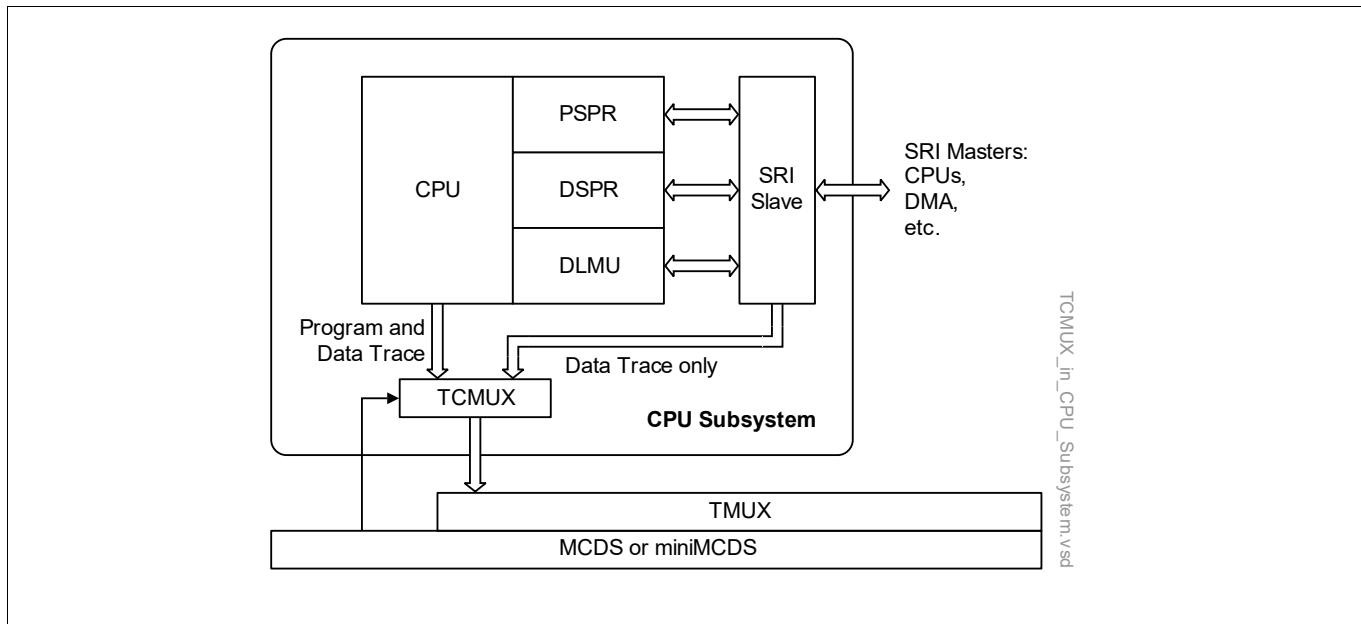
POBx, miniMCDS	POBy	POBz	BOB1	BOB2
CPU0		CPU0		
CPU1	CPU1			
CPU2	CPU2		CPU2 (TBM)	
CPU3	CPU3			CPU3 (TBM)
CPU4	CPU4		CPU4 (TBM)	
CPU5	CPU5			CPU5 (TBM)
LMU0			LMU0	
OLDA			OLDA	
				DMA_MIFO
			SPU0	SPU1
OTGB			OTGB	
OTGBM (GTM)	OTGBM (GTM)			

#### 1.4.4.2 Trace Source Multiplexer in CPU Subsystem (TCMUX)

The trace multiplexer within the CPU Subsystem ([Figure 14](#)) allows selection between:

- CPU pipeline
- PSPR/DSPR/DLMU SRI slave

*Note:* *The data trace of the SRI slave does not include the DMA channel number.*



**Figure 14 TCMUX in CPU Subsystem**

#### **1.4.4.3 Parallel Trace Use Cases**

The following use case examples are supported by the configuration with TCMUX ([Figure 14](#)) and TMUX ([Table 19](#)).

##### **Trace with miniMCDS**

In this use case all trace sources need to be selectable as input for miniMCDS. The associated first row of [Table 19](#) has only the exceptions DMA and SPU. DMA is also covered by the OTGB DMA trace options and the tracing of SRI slaves, which includes the information of DMA as master including channel number. SPU requires a continuous high bandwidth output, which miniMCDS can't provide.

##### **Trace Based Measurement (TBM)**

In this use case a parallel trace of CPUs and DMA is required. The MCDSlight provides two while the MCDS has five observation blocks . For TC39x with six CPUs plus DMA, an almost arbitrary selection from these seven sources can be made.

The data written by the remaining CPU(s) can be measured with list based DAQ, and included for instance with the traced DMA in the MCDS trace data.

##### **Debugging with MCDS**

The following combinations are mandatory for typical debug use cases:

- CPU debugging with 2-3 arbitrary CPUs + OLDA/LMU0/OTGB + DMA + SPB
- GTM debugging with OTGBM (e.g. MCS) + OTGB (e.g. signal groups) + one CPU + SPB
- OTGB + CPU(s) + SPB
- SRI slave + CPU(s)

### 1.4.5 DAP ED Interface (DAPE)

**Figure 13** shows the additional DAPE interface available on all TC3xxED and TC35x, TC3Ax devices. It has the same performance characteristics as the regular DAP interface, and it can be used in parallel to connect a second tool. Its main limitation is the access restriction to the BBB bus. **Table 20** compares DAP/DAPE in terms of features and for different applications.

**Table 20 Comparison DAP/DAPE**

Property	DAP	DAPE
Availability	on each device	TC3xxED and TC35x, TC3Ax only
Number of pins	2-3	
Performance	15-30 MB/s (both can be used in parallel)	
Package Pins	DAP/JTAG/P21	
Additional ED package mapping	no	dedicated DAPE pins on BGA packages
Accessible address ranges	all (SRI, SPB, BBB)	BBB only
Interface locking	yes (OSTATE.IF_LCK)	yes (follows OSTATE.IF_LCK)
Password exchange	yes (COMDATA)	no
Control of Application and System resets	yes (OJCONF)	no
Trigger CPU to tool sources	OTGS, TRIG	OTGS
Trigger CPU to tool signaling	TGIP, IOINFO, TRIGx	TGIP, IOINFO
CPU interrupt request	COM Mode, write SRNs	IOC32E_OJCONF
Debug with run control, flashing, etc.	yes	no
Debug with MCDS trace	yes	
Calibration	yes	no (possible with monitor)
Measurement with DAP	yes	
Measurement with AGBT	yes (DAP needed for AGBT control)	
Rapid prototyping internal	yes	yes with monitor or prolog code
Rapid prototyping external	yes	yes with service monitor

**Attention:** The additional DAPE pads used for BGA packages need to be configured with TTL level.

## 1.4.6 Revision History

**Table 21 Revision History**

Reference	Changes to Previous Version	Comment
<b>V1.0.1</b>		
-	No changes.	
<b>V1.0.2</b>		
<a href="#">Page 41, 44</a>	Description of term “Emulation Device (ED)” updated for TC3xx devices	
<a href="#">Page 43, Page 44</a>	Definitions of Cerberus and EEC updated in <a href="#">Table 18</a> and EEC part descriptions improved in paragraphs which follow General features: debug via Ethernet example removed - no functional change	
<a href="#">Page 49</a>	MCDSlight added to Trace Based Measurement use case	
<a href="#">Page 50</a>	Availability of DAPE updated	
<a href="#">Page 46</a>	New Features: Availability of Unidirectional Wide Mode UWM clarified in <a href="#">Chapter 1.4.3</a>	0000056760-78
<b>V1.0.3</b>		
<a href="#">Page 41</a>	Explanation of term “ED device”, <a href="#">TC39x/37x/33xED and TC35x, TC3Ax Comparison</a> table and requirements for TC38x and TC38xEXT code compatibility added to <a href="#">Chapter 1.4</a> .	0000048608-715
<a href="#">Page 41</a>	Name of the device TC38xEXT changed to TC3Ex in table and below in text.	0000056760-67
<b>V1.0.4</b>		
<a href="#">Page 41, Page 44, Page 50, Page 50</a>	TC3Ax added to <a href="#">Table 17, Chapter 1.4.2, Chapter 1.4.5, Table 20</a> description	0000056760-67
<a href="#">Page 41</a>	EMEM RAM size corrected for TC37xED in <a href="#">Table 17</a>	0000056760-91
<a href="#">Page 51</a>	Revision history V1.0.2 updated	
<b>V1.0.5</b>		
<a href="#">Page 41</a> to <a href="#">Page 51</a>	Due to wrong version number in footer of last release the version number has been updated to current version V1.0.5.	-

## 1.5 Software over the Air (SOTA)

### 1.5.1 Overview

All TC3xx devices besides the TC33x and TC33xED have the ability to receive Software updates Over The Air (SOTA) by providing the ability to split the PFLASH into two groups of banks, A and B. When SOTA is enabled, one of these groups of banks can be read and executed from, while the other group can have new code written to it. Thus, though simultaneous read-while-write (RWW) capability is not supported within a single physical PFLASH bank, SOTA is supported by providing the ability to safely and securely perform write and erase operations to the unused group of banks.

### 1.5.2 Functional Description

When SOTA is enabled, a group of PFLASH banks will be mapped to CPU executable address space (defined as ‘active’ banks) and the other group will be mapped to a set of addresses that allows them to be read and written to (defined as ‘inactive’ banks). **When a SOTA update has completed, and the banks are swapped around, only the address mapping will change.** This means that no data needs to be copied and the address ranges being executed from are always the same. The physical address of the PFLASH banks are as described in the standard address map in the Address Map chapter. When a SOTA address map switch from the standard address map is performed, the mapping of the PFLASH banks for read/code execution is described in the alternate address map in the Address Map chapter. In this chapter, the group of banks active in the standard address map is referred to as ‘A’ and the group of banks active in the alternate address map is referred to as ‘B’.

Note that all NVM operations are performed via the DMU using the physical system address of the PFLASH, i.e., an NVM operation always uses the standard address map regardless of swap settings. ‘NVM operation’ is a term used for any command sequence such as a program, erase etc. targeting a FLASH and does not include reads.

The parameters that control SOTA address map switching and related functions are pre-configured in UCB and the hardware configuration is only updated (by on-chip system firmware) during the subsequent System Reset. This prevents unintentional changes during application execution.

On some product variants, a 1MB block will be swapped with a 3MB block. Because the code image must be able to fit in either group A or B, the upper 2MB of the 3MB block cannot be used for program code.

#### 1.5.2.1 Performance considerations

The CPU access to its local Program Flash bank is optimised for maximum performance. This can thus cause a performance variation when executing from different physical PFLASH banks. In order to mitigate this, when SOTA is enabled, CPU fast path to local PFlash must be disabled. This will cause some drop in performance but will ensure identical system performance when executing from either groups of banks.

Another point to note is prefetch accesses. If exact performance parity between each group is required, prefetch access should be disabled completely. If however, only approximate parity is required, one of the four user assignable prefetch buffers should be assigned to each non-local CPU (the first prefetch buffer is permanently assigned to the local CPU.)

#### 1.5.2.2 Configuring for SOTA

### 1.5.2.2.1 Configuration parameters

**Table 22 Configuration Parameters related to SOTA**

Parameter	Overview Description	Copied into register (by SSW during start-up)	See Chapter
<b>SOTA Mode Enable</b>  <b>(UCB OTP.PROCONTP.SW_APEN)</b>	If valid and enabled, SOTA Mode will be entered after the next System Reset.  If valid and enabled then PROCONHSMCx and PROCONHSMCOTP setting configured for active banks will also be applied to inactive banks after the next System Reset.	DMU_HF_PROCONTP.SWAPEN  Enables SOTA bank swapping  Ensures that both group of banks have the same HSM sector protection as programmed in PROCONHSMCx/PROCONHSMCOTPx (if an HSM is present)	DMU
<b>Bank Swap</b>  <b>(UCB_SWAP_ORIG, UCB_SWAP_COPY)</b>	User programmable active address map is standard or alternate address map.  If SOTA Mode is valid and enabled and the SWAP information configured in the UCB_SWAP is valid, then after next System Reset the address map is set accordingly to standard or alternate address map.	SCU_SWAPCTRL	SCU
<b>CPUx Fast Path Disable</b>  <b>(UCB OTP_PROCONTP.CPUDDIS)</b>	Disables direct CPU access to local Program Flash Bank after the next System Reset. Access to local Program Flash routed via SRI instead.	DMU_HF_PROCONTP.DDISx  CPUx_FLASHCON4.DDIS	DMU, CPU

### 1.5.2.2.2 Initial device configuration for SOTA

Following is a recommendation for installing the initial device configuration for a device with SOTA enabled.

Starting from the delivery state, the initial execution image is programmed onto the program flash banks in the active banks. It is recommended that the sectors used are then protected by installing sector specific write protection in the UCB\_PFLASH (for this and all subsequent programming of UCBs, the standard ORIG and COPY programming sequence is to be applied. More details are available in the ‘Security’ section of DMU Chapter).

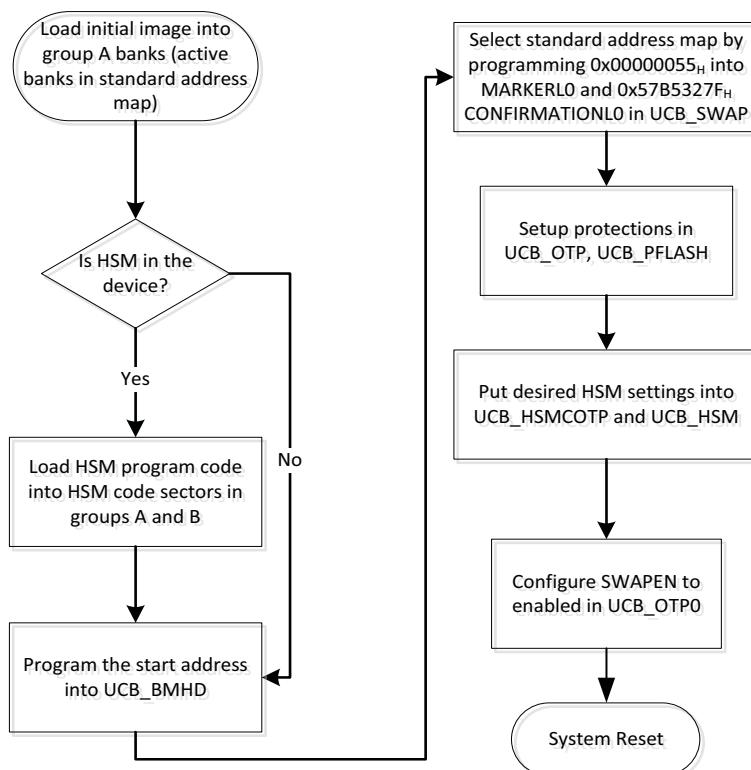
The start address of the initial image is programmed in UCB\_BMHD and the standard programming of the Boot Mode Header UCBs is performed.

In order to select the standard address map (executing from group A, writing to group B),  $00000055_{\text{H}}$  should be programmed into MARKERL0.SWAP field in UCB\_SWAP. This should then be confirmed by writing the system address of MARKERL0.SWAP into MARKERH0.ADDR, the system address of CONFIRMATIONL0.CODE into CONFIRMATIONH0.ADDR and the confirmation code  $57B5327F_{\text{H}}$  into CONFIRMATIONL0.CODE. The details of these UCB\_SWAP fields are available in the UCB Chapter located within the NVM Subsystem Chapter.

UCB OTP is programmed with the necessary values to setup the required OTP, WOP and tuning protection. Note that any OTP or WOP protected sectors cannot be re-programmed with a new image.

If HSM is required, the initial image should be loaded along with the HSM program code which should be put within the PFLASH logical sectors S0 to S39 of the first PFLASH modules of both group A and group B. The customer HSM configuration should be loaded into UCB\_HSMCOTP and UCB\_HSM. Note that any OTP protected HSM sectors cannot be re-programmed with a new image.

Finally, SWAPEN is set to enabled in UCB OTP, thus enabling SOTA mode with the next system reset.



**Figure 15 Initial SWAP configuration**

### 1.5.2.2.3 Runtime SWAP configuration

Following is a recommendation for installing the new image during running application, and configuring the device to swap to the new image.

In order to swap to the new image, first the new program image needs to be loaded into the inactive group of PFLASH banks. In order to do this, first **sector specific write protection must be disabled for those banks (by presenting the UCB\_PFLASH password for the ‘Disable Protection’ command sequence in the DMU)**. Since concurrent NVM operations (like a program or erase) to PFLASH and DFLASH is not supported, PFLASH operations must either be scheduled for times when no DFLASH operation is taking place or any ongoing DFLASH operation must be suspended to allow the PFLASH operation to occur. Thus, there needs to be some synchronization between the EEPROM driver running in the application and the Secure Flash Bootloader which is performing the update.

The newly written image should then have any errors identified and corrected before write protection is re-enabled.

**In the case of a hard failure during SOTA re-programming/erasing of PFLASH, the Replace Logical Sector feature can be used** (See DMU chapter for more details). This feature allows user to map a failing logical sector to a redundant sector using the ‘Replace Logical Sector’ command sequence.

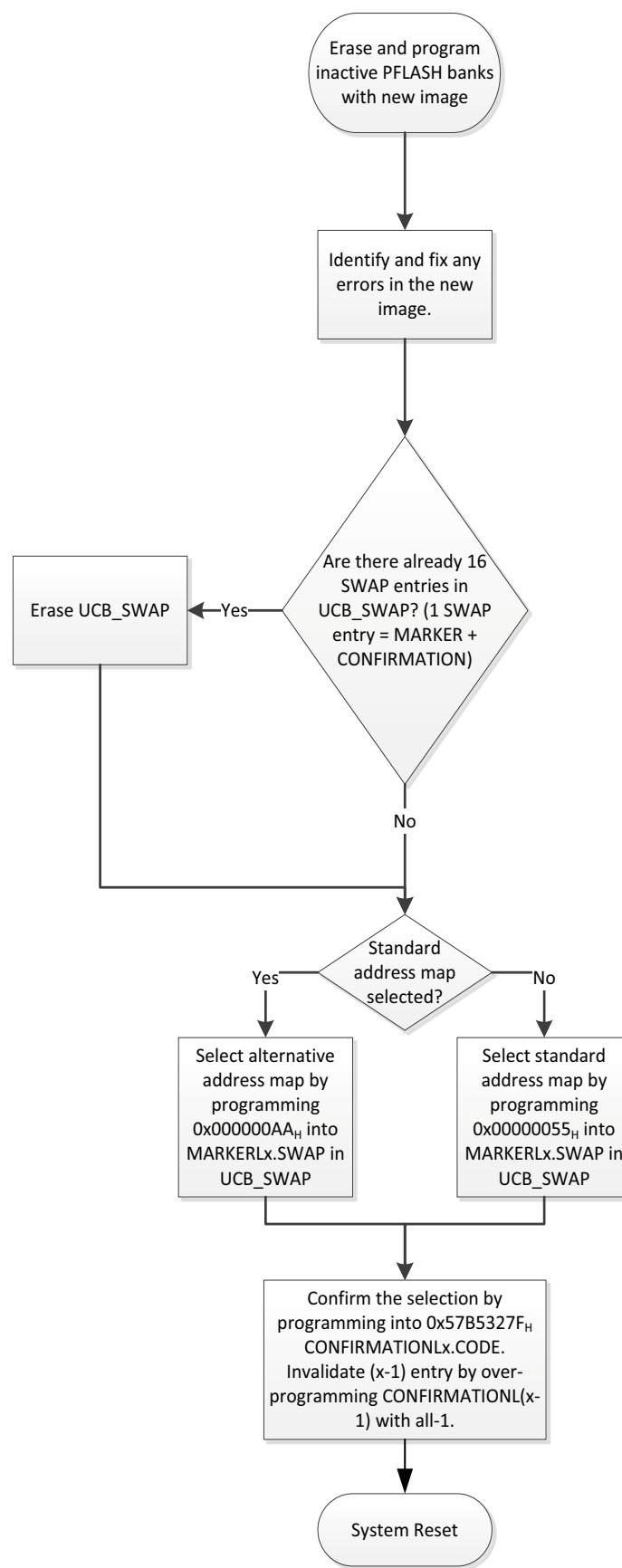
The next step is to **configure the SWAP information** (i.e., configure the address map to be selected) in the **UCB\_SWAP**. After the UCB\_SWAP password has been presented (using the 'Disable Protection' DMU command sequence), MARKERLx.SWAP is changed to  $000000AA_H$  (thus selecting alternate address map) if group B contains the **new image** or  $00000055_H$  (thus selecting standard address map) if group A does. **MARKERHx.ADDR**, **CONFIRMATIONHx.ADDR** and **CONFIRMATIONLx.CODE** is then programmed as with the initial configuration. The previous (x-1) UCB\_SWAP entry is invalidated by over-programming all-1 into CONFIRMATIONL(x-1) and CONFIRMATIONH(x-1) (over-programming with all-1 delivers an ECC correct result for UCB and DFLASH). For all of these, 'x' should be increased by one each time images are swapped, starting from 1 the first time after the initial configuration.

If UCB\_SWAP is full (i.e., 'x' has reached 15), the whole UCB may be erased and 'x' set back to 0 before a new entry is added. The write protection is then re-installed by using the 'Resume Protection' DMU command sequence.

Note that the last valid entry of the SWAP information (i.e., SWAP information stored at the highest value of 'x') is used by the Startup Software for configuring SOTA in the system.

Thus, it is possible to configure 16 SWAPs in the UCB\_SWAP before an erase is required. The maximum number of SWAPs possible during the lifetime of the device is dependent on the Datasheet parameter for PFLASH erase/program cycles ( $N_{E\_P} = 1000$  cycles). In order to perform 1000 SWAP configurations, atleast a total of 124 UCB erase/program cycles is required during the lifetime (two UCB erases required per 16 SWAP updates as both UCB\_SWAP\_ORIG and UCB\_SWAP\_COPY needs to be erased and updated). This needs to be taken into account while updating the other UCBs in order to adhere to the Datasheet parameter for UCB program/erase cycles ( $t_{RTU} = 400$  erase/program cycles for all UCBs together).

To begin running of the new image, a system reset should be triggered (application resets have no effect).

**Figure 16 Runtime SOTA configuration**

### 1.5.3 Safety

If SOTA is disabled, the entirety of the program flash is protected by safety\_endinit which prevents unintentional changes to the program flash's contents. If SOTA is enabled, safety\_endinit protection for the inactive group of banks is automatically removed, thus allowing them to be updated. However, even for the active banks, the requirement for safety application software to be checked before execution remains. More details on the safety\_endinit protection are available in the 'Functional Safety Features' section of the NVM Subsystem chapter.

### 1.5.4 Security

The security protection offered to a NVM operation on the PFLASH remains the same as defined in the 'Security' section of the DMU Chapter irrespective of the active or inactive nature of the PFLASH bank.

There is however, additional measures implemented for handling HSM Exclusive sectors in PFLASH.

If SOTA is enabled, any protection configured in the PROCONHSMCX and PROCONHSMCOTP registers is mirrored to the PFLASH logical sectors S0 to S39 of both the groups A and B. This is to prevent unauthorised access to HSM code being gained via bank swapping. The user must make sure that the secure content image is duplicated between the two groups. Reprogramming sectors marked as HSM exclusive, even when inactive, can only be done by the HSM or Cerberus when HSM debug is enabled.

### 1.5.5 Revision History

**Table 23 Revision History**

Reference	Change to Previous Version	Change Request Comment
<b>V1.0.1</b>		
	Revision History added	
	First-level heading structure created	
	Restructuring of the chapter for better clarity, and addition of programming hints, and details to support SOTA implementation in both hardware and software.	
<b>Chapter 1.</b> <b>5.2.2.2</b>	Clarification of the address to be stored in MARKERH0.ADDR and CONFIRMATIONH0.ADDR	
<b>V1.0.2</b>		
<b>Chapter 1.</b> <b>5.2.2.2</b>	Removed UCB_HSMCFG from the user programming model.	Jira 0000049147-100
<b>Chapter 1.</b> <b>5.2.2.3</b>	Added recommendation to invalidate the previous entry by over-programming with all-1.	Jira 0000052718-112
<b>V1.0.3</b>		
<b>Page 52</b>	<b>Chapter 1.5.2</b> - Typo when referring to B image - should be banks rather than maps.	Jira 0000056754-43
<b>V1.0.4</b>		
<b>Page 55</b>	<b>Chapter 1.5.2.2.3</b> - Due to 64 bit width of page write for UCBs, must over-programming CONFIRMATIONL+H with all-1.	Jira 0000056760-70

## Memory Maps (MEMMAP)

## 2 Memory Maps (MEMMAP)

The MEMMAP specifies the over-arching memory map for all devices in the AURIX TC3xx product family.

The device specific MEMMAP (reflecting the exact silicon content) are specified in the device Appendix books.

### 2.1 Feature List

#### FEATURE LIST

The address map includes the following memories:

- Program Flash Interface (PFI):
  - Program Flash Memory (PF)
- Data Memory Unit (DMU):
  - Data Flash Memory for CPU EEPROM (DF0)
  - User Configuration Blocks (DF0)
  - Configuration Sector (DF0)
  - Data Flash Memory for HSM EEPROM (DF1)
- CPU0 and CPU1:
  - 64 Kbyte of Program Scratch-Pad SRAM (PSPR)<sup>1)</sup>
  - 240 Kbyte of Data Scratch-Pad SRAM (DSPR)<sup>1)</sup>
  - 32 Kbyte of Program Cache (P-Cache)
  - 16 Kbyte of Data Cache (D-Cache)
  - 64 Kbyte of Local Memory Unit (DLMU)
- CPU2 to CPU5:
  - 64 Kbyte of Program Scratch-Pad SRAM (PSPR)<sup>1)</sup>
  - 96 Kbyte of Data Scratch-Pad SRAM (DSPR)<sup>1)</sup>
  - 32 Kbyte of Program Cache (P-Cache)
  - 16 Kbyte of Data Cache (D-Cache)
  - 64 Kbyte of Local Memory Unit (DLMU)
- Local Memory Unit (LMU):
  - LMU SRAM (CPU DLMU or LMU LMURAM)<sup>1)</sup>
  - DAM SRAM (DAMRAM)
- Boot ROM (BROM)

Furthermore, the device has the following on-chip buses:

- System Peripheral Bus (SPB)
- Shared Resource Interconnect (SRI)
- Back Bone Bus (BBB)

### 2.2 Overview

The memory map describes the address locations and access possibilities for the units, memories, and reserved areas as “seen” from the different on-chip buses’ point of view.

<sup>1)</sup> Before used by the application, the memory has to be initialized (see chapter ‘Memory Test Unit’, MTU)

## Memory Maps (MEMMAP)

### 2.3 Functional Description

The bus-specific address maps describe how the different bus master devices react on accesses to on-chip memories and modules, and which address ranges are valid or invalid for the corresponding buses.

The detailed address mapping of e.g. control registers, SRAM blocks or flash banks/sectors within a module is described in the related module chapter.

**Note:** *In addition to the here described system address map, each TriCore has a TriCore IP internal access to its PSPR via C000\_0000<sub>H</sub> and an internal access to its DSPR via D000\_0000<sub>H</sub>. This additional/private view to the local scratch pad SRAMs is described in the CPU chapter.*

**Table 24** defines the acronyms and other terms that are used in the address maps (**Table 25** and **Table 41**).

**Table 24 Definition of Acronyms and Terms**

Term	Description
BBBEE	A bus access is terminated with a bus error on the BBB.
SPBEE	A bus access is terminated with a bus error on the SPB.
SRIE	A bus access is terminated with a bus error on the SRI.
Access	A bus access is allowed and is executed.

#### 2.3.1 Segments

This section summarizes the contents of the segments.

##### Segments 0 and 2

These memory segments are reserved.

##### Segments 1 and 3-7

These memory segments allow access to the CPUs Program and Data Scratch Pad SRAM (PSPR, DSPR), Program and Data Cache SRAMs (PCACHE, DCACHE) as well as TAG SRAMs related to Program and Data Cache (PTAG SRAM<sup>1)</sup> and DTAG SRAM<sup>1)</sup>).

Where DCACHE is supported, DCACHE and DTAG SRAM<sup>1)</sup> can be only accessed if the Data Cache is disabled.

PCACHE and PTAG SRAMs<sup>1)</sup> can be only accessed if the related Program Cache is disabled.

The attribute of these segments (cached / non-cached) can be partially configured<sup>2)</sup> for each CPUs data and program side individually (see CPU chapter: Physical Memory Attribute Registers, PMAX).

##### Segment 8

This memory segment allows cached access to PFlash and BROM.

##### Segment 9

This memory segment allows cached access to LMU and to EMEM.

##### Segment 10

This memory segment allows non-cached access to PFlash, DFlash and BROM.

1) TAG SRAMs are not meant to be used as general SRAMs and can be accessed only with single data access and only with 64 bit aligned address.

2) Mapping of Cache and TAG SRAMs is controlled via the MTU register MTU\_MEMMAP.

## Memory Maps (MEMMAP)

### Segment 11

This memory segment allows non-cached access to LMU and to EMEM.

### Segment 12

This memory segment is reserved.

### Segment 13

This memory segment is reserved.

### Segment 14

This memory segment is reserved.

### Segment 15

The lower 128 Mbyte is SPB address space and the upper 128 Mbyte is SRI address space.

## 2.3.2 Address Map of the On Chip Bus System

All bus master agents can address identical peripherals and memories at identical addresses. The system address map is visible and valid for all CPUs which means that all peripherals and resources are accessible from all TriCore CPUs and other on chip bus master agents.

Parallel access by more than one bus master agent to one slave agent are executed sequentially. Additionally the SRI, SPB and BBB support atomic Read Modify Write sequences from the CPUs.

### 2.3.2.1 Segments 0 to 14

**Table 25** shows the address map of segments 0 to 14.

#### Notes

1. *Write Access Type:* Write access to Flash resources are handled by the DMU module (Flash command sequence, see DMU chapter for details).

**Table 25 Address Map of Segment 0 to 14**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
0	0000 0000 <sub>H</sub> - 0000 0007 <sub>H</sub>	8 Byte	Reserved (virtual address space)	SRIBE / SPBBE <sup>1)</sup>	SRIBE / SPBBE <sup>1)</sup>
	0000 0008 <sub>H</sub> - 0FFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
1	1000 0000 <sub>H</sub> - 1001 7FFF <sub>H</sub>	96 Kbyte	CPU5 Data Scratch-Pad SRAM (CPU5 DSPR)	Access	Access
	1001 8000 <sub>H</sub> - 1001 BFFF <sub>H</sub>	16 Kbyte	CPU5. Data Cache SRAM (CPU5 DCACHE)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	1001 C000 <sub>H</sub> - 100B FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	100C 0000 <sub>H</sub> - 100C 17FF <sub>H</sub>	-	CPU5 Data Cache TAG SRAM <sup>3)</sup> (CPU5 DTAG)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	100C 1800 <sub>H</sub> - 100F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Memory Maps (MEMMAP)****Table 25 Address Map of Segment 0 to 14 (cont'd)**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
1	1010 0000 <sub>H</sub> - 1010 FFFF <sub>H</sub>	64 Kbyte	CPU5 Program Scratch-Pad SRAM (CPU5 PSPR)	Access	Access
	1011 0000 <sub>H</sub> - 1011 7FFF <sub>H</sub>	32 Kbyte	CPU5.Program Cache SRAM (CPU5 PCACHE)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	1011 8000 <sub>H</sub> - 101B FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	101C 0000 <sub>H</sub> - 101C 2FFF <sub>H</sub>	-	CPU5 Program Cache TAG SRAM <sup>3)</sup> (CPU5 PTAG)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	101C 3000 <sub>H</sub> - 1FFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
2	2000 0000 <sub>H</sub> - 2FFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
3	3000 0000 <sub>H</sub> - 3001 7FFF <sub>H</sub>	96 Kbyte	CPU4 Data Scratch-Pad SRAM (CPU4 DSPR)	Access	Access
	3001 8000 <sub>H</sub> - 3001 BFFF <sub>H</sub>	16 Kbyte	CPU4. Data Cache SRAM (CPU4 DCACHE)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	3001 C000 <sub>H</sub> - 300B FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	300C 0000 <sub>H</sub> - 300C 17FF <sub>H</sub>	-	CPU4 Data Cache TAG SRAM <sup>1)</sup> (CPU4 DTAG)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	300C 1800 <sub>H</sub> - 300F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	3010 0000 <sub>H</sub> - 3010 FFFF <sub>H</sub>	64 Kbyte	CPU4 Program Scratch-Pad SRAM (CPU4 PSPR)	Access	Access
	3011 0000 <sub>H</sub> - 3011 7FFF <sub>H</sub>	32 Kbyte	CPU4.Program Cache SRAM (CPU4 PCACHE)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	3011 8000 <sub>H</sub> - 301B FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	301C 0000 <sub>H</sub> - 301C 2FFF <sub>H</sub>	-	CPU4 Program Cache TAG SRAM <sup>1)</sup> (CPU4 PTAG)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	301C 3000 <sub>H</sub> - 3FFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
4	4000 0000 <sub>H</sub> - 4001 7FFF <sub>H</sub>	96 Kbyte	CPU3 Data Scratch-Pad SRAM (CPU3 DSPR)	Access	Access
	4001 8000 <sub>H</sub> - 4001 BFFF <sub>H</sub>	16 Kbyte	CPU3. Data Cache SRAM (CPU3 DCACHE)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	4001 C000 <sub>H</sub> - 400B FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	400C 0000 <sub>H</sub> - 400C 17FF <sub>H</sub>	-	CPU3 Data Cache TAG SRAM <sup>1)</sup> (CPU3 DTAG)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	400C 1800 <sub>H</sub> - 400F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	4010 0000 <sub>H</sub> - 4010 FFFF <sub>H</sub>	64 Kbyte	CPU3 Program Scratch-Pad SRAM (CPU3 PSPR)	Access	Access
	4011 0000 <sub>H</sub> - 4011 7FFF <sub>H</sub>	32 Kbyte	CPU3.Program Cache SRAM (CPU3 PCACHE)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	4011 8000 <sub>H</sub> - 401B FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	401C 0000 <sub>H</sub> - 401C 2FFF <sub>H</sub>	-	CPU3 Program Cache TAG SRAM <sup>1)</sup> (CPU3 PTAG)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE

**Memory Maps (MEMMAP)****Table 25 Address Map of Segment 0 to 14 (cont'd)**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
5	401C 3000 <sub>H</sub> - 4FFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	5000 0000 <sub>H</sub> - 5001 7FFF <sub>H</sub>	96 Kbyte	CPU2 Data Scratch-Pad SRAM (CPU2 DSPR)	Access	Access
	5001 8000 <sub>H</sub> - 5001 BFFF <sub>H</sub>	16 Kbyte	CPU2. Data Cache SRAM (CPU2 DCACHE)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	5001 C000 <sub>H</sub> - 500B FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	500C 0000 <sub>H</sub> - 500C 17FF <sub>H</sub>	-	CPU2 Data Cache TAG SRAM <sup>1)</sup> (CPU2 DTAG)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	500C 1800 <sub>H</sub> - 500F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	5010 0000 <sub>H</sub> - 5010 FFFF <sub>H</sub>	64 Kbyte	CPU2 Program Scratch-Pad SRAM (CPU2 PSPR)	Access	Access
	5011 0000 <sub>H</sub> - 5011 7FFF <sub>H</sub>	32 Kbyte	CPU2.Program Cache SRAM (CPU2 PCACHE)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	5011 8000 <sub>H</sub> - 501B FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	501C 0000 <sub>H</sub> - 501C 2FFF <sub>H</sub>	-	CPU2 Program Cache TAG SRAM <sup>1)</sup> (CPU2 PTAG)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	501C 3000 <sub>H</sub> - 5FFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
6	6000 0000 <sub>H</sub> - 6003 BFFF <sub>H</sub>	240 Kbyte	CPU1 Data Scratch-Pad SRAM (CPU1 DSPR)	Access	Access
	6003 C000 <sub>H</sub> - 6003 FFFF <sub>H</sub>	16 Kbyte	CPU1. Data Cache SRAM (CPU1 DCACHE)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	6004 0000 <sub>H</sub> - 600B FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	600C 0000 <sub>H</sub> - 600C 17FF <sub>H</sub>	-	CPU1 Data Cache TAG SRAM <sup>1)</sup> (CPU1 DTAG)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	600C 1800 <sub>H</sub> - 600F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	6010 0000 <sub>H</sub> - 6010 FFFF <sub>H</sub>	64 Kbyte	CPU1 Program Scratch-Pad SRAM (CPU1 PSPR)	Access	Access
	6011 0000 <sub>H</sub> - 6011 7FFF <sub>H</sub>	32 Kbyte	CPU1.Program Cache SRAM (CPU1 PCACHE)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	6011 8000 <sub>H</sub> - 601B FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	601C 0000 <sub>H</sub> - 601C 2FFF <sub>H</sub>	-	CPU1 Program Cache TAG SRAM <sup>1)</sup> (CPU1 PTAG)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	601C 3000 <sub>H</sub> - 6FFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
7	7000 0000 <sub>H</sub> - 7003 BFFF <sub>H</sub>	240 Kbyte	CPU0 Data Scratch-Pad SRAM (CPU0 DSPR)	Access	Access
	7003 C000 <sub>H</sub> - 7003 FFFF <sub>H</sub>	16 Kbyte	CPU0. Data Cache SRAM (CPU0 DCACHE)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	7004 0000 <sub>H</sub> - 700B FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	700C 0000 <sub>H</sub> - 700C 17FF <sub>H</sub>	-	CPU0 Data Cache TAG SRAM <sup>1)</sup> (CPU0 DTAG)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE

**Memory Maps (MEMMAP)****Table 25 Address Map of Segment 0 to 14 (cont'd)**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
	700C 1800 <sub>H</sub> - 700F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	7010 0000 <sub>H</sub> - 7010 FFFF <sub>H</sub>	64 Kbyte	CPU0 Program Scratch-Pad SRAM (CPU0 PSPR)	Access	Access
	7011 0000 <sub>H</sub> - 7011 7FFF <sub>H</sub>	32 Kbyte	CPU0.Program Cache SRAM (CPU0 PCACHE)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	7011 8000 <sub>H</sub> - 701B FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	701C 0000 <sub>H</sub> - 701C 2FFF <sub>H</sub>	-	CPU0 Program Cache TAG SRAM <sup>1)</sup> (CPU0 PTAG)	Access <sup>2)</sup> / SRIBE	Access <sup>2)</sup> / SRIBE
	701C 3000 <sub>H</sub> - 7FFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
8	8000 0000 <sub>H</sub> - 802F FFFF <sub>H</sub>	3 Mbyte	Program Flash 0 (PF0)	Access	SRIBE
	8030 0000 <sub>H</sub> - 805F FFFF <sub>H</sub>	3 Mbyte	Program Flash 1 (PF1)	Access	SRIBE
	8060 0000 <sub>H</sub> - 808F FFFF <sub>H</sub>	3 Mbyte	Program Flash 2 (PF2)	Access	SRIBE
	8090 0000 <sub>H</sub> - 80BF FFFF <sub>H</sub>	3 Mbyte	Program Flash 3 (PF3)	Access	SRIBE
	80C0 0000 <sub>H</sub> - 80EF FFFF <sub>H</sub>	3 Mbyte	Program Flash 4 (PF4)	Access	SRIBE
	80F0 0000 <sub>H</sub> - 80FF FFFF <sub>H</sub>	1 Mbyte	Program Flash 5 (PF5)	Access	SRIBE
	8100 0000 <sub>H</sub> - 811F FFFF <sub>H</sub>	2 Mbyte	Reserved (for PFLASH)	SRIBE	SRIBE
	8120 0000 <sub>H</sub> - 81FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	8200 0000 <sub>H</sub> - 87FF FFFF <sub>H</sub>	96 Mbyte	External Bus Unit (EBU)	Access	Access
	8800 0000 <sub>H</sub> - 8FDF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	8FE0 0000 <sub>H</sub> - 8FE7 FFFF <sub>H</sub>	512 Kbyte	Online Data Acquisition (OLDA)	SRIBE	Access / SRIBE
	8FE8 0000 <sub>H</sub> - 8FFE FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	8FFF 0000 <sub>H</sub> - 8FFF FFFF <sub>H</sub>	64 Kbyte	Boot ROM (BROM)	Access	SRIBE
9	9000 0000 <sub>H</sub> - 9000 FFFF <sub>H</sub>	64 Kbyte	LMU (CPU0 DLMU)	Access	Access
	9001 0000 <sub>H</sub> - 9001 FFFF <sub>H</sub>	64 Kbyte	LMU (CPU1 DLMU)	Access	Access
	9002 0000 <sub>H</sub> - 9002 FFFF <sub>H</sub>	64 Kbyte	LMU (CPU2 DLMU)	Access	Access
	9003 0000 <sub>H</sub> - 9003 FFFF <sub>H</sub>	64 Kbyte	LMU (CPU3 DLMU)	Access	Access
	9004 0000 <sub>H</sub> - 9007 FFFF <sub>H</sub>	256 Kbyte	LMU (LMU0 LMURAM)	Access	Access
	9008 0000 <sub>H</sub> - 900B FFFF <sub>H</sub>	256 Kbyte	LMU (LMU1 LMURAM)	Access	Access
	900C 0000 <sub>H</sub> - 900F FFFF <sub>H</sub>	256 Kbyte	LMU (LMU2 LMURAM)	Access	Access
	9010 0000 <sub>H</sub> - 9010 FFFF <sub>H</sub>	64 Kbyte	LMU (CPU4 DLMU)	Access	Access
	9011 0000 <sub>H</sub> - 9011 FFFF <sub>H</sub>	64 Kbyte	LMU (CPU5 DLMU)	Access	Access
	9012 0000 <sub>H</sub> - 903F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	9040 0000 <sub>H</sub> - 9040 7FFF <sub>H</sub>	32 Kbyte	DAM (DAM0 RAM0)	Access	Access
	9040 8000 <sub>H</sub> - 9040 FFFF <sub>H</sub>	32 Kbyte	DAM (DAM0 RAM1)	Access	Access
	9041 0000 <sub>H</sub> - 9041 7FFF <sub>H</sub>	32 Kbyte	DAM (DAM1 RAM0)	Access	Access
	9041 8000 <sub>H</sub> - 9041 FFFF <sub>H</sub>	32 Kbyte	DAM (DAM1 RAM1)	Access	Access
	9042 0000 <sub>H</sub> - 97FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

## Memory Maps (MEMMAP)

Table 25 Address Map of Segment 0 to 14 (cont'd)

Segment	Address Range	Size	Description	Access Type	
				Read	Write
	9800 0000 <sub>H</sub> - 9800 1FFF <sub>H</sub>	8 Kbyte	MINIMCDS Trace SRAM (TRAM)	Access <sup>4)</sup>	Access <sup>4)</sup>
	9800 2000 <sub>H</sub> - 98DF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	98E0 0000 <sub>H</sub> - 98EF FFFF <sub>H</sub>	1 Mbyte	EMEM (EMEM Module 4)	Access	Access
	98F0 0000 <sub>H</sub> - 98FF FFFF <sub>H</sub>	1 Mbyte	EMEM (EMEM Module 5)	Access	Access
	9900 0000 <sub>H</sub> - 990F FFFF <sub>H</sub>	1 Mbyte	EMEM (EMEM Module 0)	Access	Access
	9910 0000 <sub>H</sub> - 991F FFFF <sub>H</sub>	1 Mbyte	EMEM (EMEM Module 1)	Access	Access
	9920 0000 <sub>H</sub> - 992F FFFF <sub>H</sub>	1 Mbyte	EMEM (EMEM Module 2)	Access	Access
	9930 0000 <sub>H</sub> - 993F FFFF <sub>H</sub>	1 Mbyte	EMEM (EMEM Module 3)	Access	Access
	9940 0000 <sub>H</sub> - 9FFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
10	A000 0000 <sub>H</sub> - A02F FFFF <sub>H</sub>	3 Mbyte	Program Flash 0 (PF0)	Access	SRIBE
	A030 0000 <sub>H</sub> - A05F FFFF <sub>H</sub>	3 Mbyte	Program Flash 1 (PF1)	Access	SRIBE
	A060 0000 <sub>H</sub> - A08F FFFF <sub>H</sub>	3 Mbyte	Program Flash 2 (PF2)	Access	SRIBE
	A090 0000 <sub>H</sub> - A0BF FFFF <sub>H</sub>	3 Mbyte	Program Flash 3 (PF3)	Access	SRIBE
	A0C0 0000 <sub>H</sub> - A0EF FFFF <sub>H</sub>	3 Mbyte	Program Flash 4 (PF4)	Access	SRIBE
	A0F0 0000 <sub>H</sub> - A0FF FFFF <sub>H</sub>	1 Mbyte	Program Flash 5 (PF5)	Access	SRIBE
	A100 0000 <sub>H</sub> - A11F FFFF <sub>H</sub>	2 Mbyte	Reserved (for PFLASH)	SRIBE	SRIBE
	A120 0000 <sub>H</sub> - A1FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A200 0000 <sub>H</sub> - A7FF FFFF <sub>H</sub>	96 Mbyte	External Bus Unit (EBU)	Access	Access
	A800 0000 <sub>H</sub> - A800 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 0 (EC0)	Access	SRIBE
	A800 4000 <sub>H</sub> - A807 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A808 0000 <sub>H</sub> - A80B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 0 (PFI0)	Access	SRIBE
	A80C 0000 <sub>H</sub> - A82F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A830 0000 <sub>H</sub> - A830 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 1 (EC1)	Access	SRIBE
	A830 4000 <sub>H</sub> - A837 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A838 0000 <sub>H</sub> - A83B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 1 (PFI1)	Access	SRIBE
	A83C 0000 <sub>H</sub> - A85F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A860 0000 <sub>H</sub> - A860 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 2 (EC2)	Access	SRIBE
	A860 4000 <sub>H</sub> - A867 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A868 0000 <sub>H</sub> - A86B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 2 (PFI2)	Access	SRIBE
	A86C 0000 <sub>H</sub> - A88F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A890 0000 <sub>H</sub> - A890 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 3 (EC3)	Access	SRIBE
	A890 4000 <sub>H</sub> - A897 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A898 0000 <sub>H</sub> - A89B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 3 (PFI3)	Access	SRIBE
	A89C 0000 <sub>H</sub> - A8BF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A8C0 0000 <sub>H</sub> - A8C0 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 4 (EC4)	Access	SRIBE
	A8C0 4000 <sub>H</sub> - A8C7 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A8C8 0000 <sub>H</sub> - A8CB FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 4 (PFI4)	Access	SRIBE

## Memory Maps (MEMMAP)

Table 25 Address Map of Segment 0 to 14 (cont'd)

Segment	Address Range	Size	Description	Access Type	
				Read	Write
	A8CC 0000 <sub>H</sub> - A8EF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A8F0 0000 <sub>H</sub> - A8F0 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 5 (EC5)	Access	SRIBE
	A8F0 4000 <sub>H</sub> - A8F7 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A8F8 0000 <sub>H</sub> - A8FB FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 5 (PFI5)	Access	SRIBE
	A8FC 0000 <sub>H</sub> - AEFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	AF00 0000 <sub>H</sub> - AF0F FFFF <sub>H</sub>	1 Mbyte	Data Flash 0 EEPROM (DF0) Host Comd. Sequence Interpreter	Access	Access <sup>5)</sup>
	AF10 0000 <sub>H</sub> - AF3F FFFF <sub>H</sub>	3 Mbyte	Reserved	SRIBE	SRIBE
	AF40 0000 <sub>H</sub> - AF40 5FFF <sub>H</sub>	24 Kbyte	Data Flash 0 UCB (DF0)	Access	SRIBE
	AF40 6000 <sub>H</sub> - AF7F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	AF80 0000 <sub>H</sub> - AF80 FFFF <sub>H</sub>	64 Kbyte	Data Flash 0 CFS (DF0)	Access	SRIBE
	AF81 0000 <sub>H</sub> - AFBF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	AFC0 0000 <sub>H</sub> - AFC1 FFFF <sub>H</sub>	128 Kbyte	Data Flash 1 EEPROM (DF1) HSM Comd. Sequence Interpreter	Access	Access <sup>6)</sup>
	AFC2 0000 <sub>H</sub> - AFC3 FFFF <sub>H</sub>	128 Kbyte	Reserved	SRIBE	SRIBE
	AFC4 0000 <sub>H</sub> - AFDF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	AFE0 0000 <sub>H</sub> - AFE7 FFFF <sub>H</sub>	512 Kbyte	Online Data Acquisition (OLDA)	SRIBE	Access / SRIBE
	AFE8 0000 <sub>H</sub> - AFFE FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	AFFF 0000 <sub>H</sub> - AFFF FFFF <sub>H</sub>	64 Kbyte	Boot ROM (BROM)	Access	SRIBE
11	B000 0000 <sub>H</sub> - B000 FFFF <sub>H</sub>	64 Kbyte	LMU (CPU0 DLMU)	Access	Access
	B001 0000 <sub>H</sub> - B001 FFFF <sub>H</sub>	64 Kbyte	LMU (CPU1 DLMU)	Access	Access
	B002 0000 <sub>H</sub> - B002 FFFF <sub>H</sub>	64 Kbyte	LMU (CPU2 DLMU)	Access	Access
	B003 0000 <sub>H</sub> - B003 FFFF <sub>H</sub>	64 Kbyte	LMU (CPU3 DLMU)	Access	Access
	B004 0000 <sub>H</sub> - B007 FFFF <sub>H</sub>	256 Kbyte	LMU (LMU0 LMURAM)	Access	Access
	B008 0000 <sub>H</sub> - B00B FFFF <sub>H</sub>	256 Kbyte	LMU (LMU1 LMURAM)	Access	Access
	B00C 0000 <sub>H</sub> - B00F FFFF <sub>H</sub>	256 Kbyte	LMU (LMU2 LMURAM)	Access	Access
	B010 0000 <sub>H</sub> - B010 FFFF <sub>H</sub>	64 Kbyte	LMU (CPU4 DLMU)	Access	Access
	B011 0000 <sub>H</sub> - B011 FFFF <sub>H</sub>	64 Kbyte	LMU (CPU5 DLMU)	Access	Access
	B012 0000 <sub>H</sub> - B03F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	B040 0000 <sub>H</sub> - B040 7FFF <sub>H</sub>	32 Kbyte	DAM (DAM0 RAM0)	Access	Access
	B040 8000 <sub>H</sub> - B040 FFFF <sub>H</sub>	32 Kbyte	DAM (DAM0 RAM1)	Access	Access
	B041 0000 <sub>H</sub> - B041 7FFF <sub>H</sub>	32 Kbyte	DAM (DAM1 RAM0)	Access	Access
	B041 8000 <sub>H</sub> - B041 FFFF <sub>H</sub>	32 Kbyte	DAM (DAM1 RAM1)	Access	Access
	B042 0000 <sub>H</sub> - B7FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	B800 0000 <sub>H</sub> - B800 1FFF <sub>H</sub>	8 Kbyte	MINIMCDS Trace SRAM (TRAM)	Access <sup>4)</sup>	Access <sup>4)</sup>
	B800 2000 <sub>H</sub> - B8DF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Memory Maps (MEMMAP)****Table 25 Address Map of Segment 0 to 14 (cont'd)**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
	B8E0 0000 <sub>H</sub> - B8EF FFFF <sub>H</sub>	1 Mbyte	EMEM (EMEM Module 4)	Access	Access
	B8F0 0000 <sub>H</sub> - B8FF FFFF <sub>H</sub>	1 Mbyte	EMEM (EMEM Module 5)	Access	Access
	B900 0000 <sub>H</sub> - B90F FFFF <sub>H</sub>	1 Mbyte	EMEM (EMEM Module 0)	Access	Access
	B910 0000 <sub>H</sub> - B91F FFFF <sub>H</sub>	1 Mbyte	EMEM (EMEM Module 1)	Access	Access
	B920 0000 <sub>H</sub> - B92F FFFF <sub>H</sub>	1 Mbyte	EMEM (EMEM Module 2)	Access	Access
	B930 0000 <sub>H</sub> - B93F FFFF <sub>H</sub>	1 Mbyte	EMEM (EMEM Module 3)	Access	Access
	B940 0000 <sub>H</sub> - B947 FFFF <sub>H</sub>	512 Kbyte	Extra Trace Memory (XTM) (only 16 Kbyte physical SRAM)	Access	Access
	B948 0000 <sub>H</sub> - BFFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
12	C000 0000 <sub>H</sub> - CFFF FFFF <sub>H</sub>	-	Reserved <sup>7)</sup>	SRIBE	SRIBE
13	D000 0000 <sub>H</sub> - DFFF FFFF <sub>H</sub>	-	Reserved <sup>7)</sup>	SRIBE	SRIBE
14	E000 0000 <sub>H</sub> - EFFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
15	F000 0000 <sub>H</sub> - FFFF FFFF <sub>H</sub>	256 Mbyte	See <a href="#">Table 41</a>		

- 1) If an SPB access to 0000 0000H occurs, the SPB BCU generates a bus error.
- 2) PCACHE/DCACHE SRAMs (and the corresponding TAG SRAMs) can be only accessed when mapped into the address space (PCACHE / DCACHE disabled. See CPU chapter, register SMACON for details).
- 3) TAG SRAMs are not meant to be used as general SRAMs and can be accessed only with single data access and only with 64 bit aligned address. Mapping of TAG SRAMs in the address map can be used as additional option for memory testing
- 4) TRAM shall not be used as a general SRAM and can only be accessed when OCDS is enabled.
- 5) Host Command Sequence Interpreter
- 6) HSM Command Sequence Interpreter
- 7) See also chapter 'CPU, 'Local and Global Addressing' for CPU local views to segment 'C' and segment 'D'.

**Table 26 TC39x Alternate Address Map for SOTA of Segment 8 PFLASH**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
8	8000 0000 <sub>H</sub> - 802F FFFF <sub>H</sub>	3 Mbyte	Program Flash 2 (PF2)	Access	SRIBE
	8030 0000 <sub>H</sub> - 805F FFFF <sub>H</sub>	3 Mbyte	Program Flash 3 (PF3)	Access	SRIBE
	8060 0000 <sub>H</sub> - 808F FFFF <sub>H</sub>	3 Mbyte	Program Flash 0 (PF0)	Access	SRIBE
	8090 0000 <sub>H</sub> - 80BF FFFF <sub>H</sub>	3 Mbyte	Program Flash 1 (PF1)	Access	SRIBE
	80C0 0000 <sub>H</sub> - 80CF FFFF <sub>H</sub>	1 Mbyte	Program Flash 5 (PF5)	Access	SRIBE
	80D0 0000 <sub>H</sub> - 80EF FFFF <sub>H</sub>	2 Mbyte	Reserved (for PFLASH)	SRIBE	SRIBE
	80F0 0000 <sub>H</sub> - 811F FFFF <sub>H</sub>	3 Mbyte	Program Flash 4 (PF4)	Access	SRIBE
	8120 0000 <sub>H</sub> - 81FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Memory Maps (MEMMAP)****Table 27 TC39x Alternate Address Map for SOTA of Segment 10 PFLASH**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
10	A000 0000 <sub>H</sub> - A02F FFFF <sub>H</sub>	3 Mbyte	Program Flash 2 (PF2)	Access	SRIBE
	A030 0000 <sub>H</sub> - A05F FFFF <sub>H</sub>	3 Mbyte	Program Flash 3 (PF3)	Access	SRIBE
	A060 0000 <sub>H</sub> - A08F FFFF <sub>H</sub>	3 Mbyte	Program Flash 0 (PF0)	Access	SRIBE
	A090 0000 <sub>H</sub> - A0BF FFFF <sub>H</sub>	3 Mbyte	Program Flash 1 (PF1)	Access	SRIBE
	A0C0 0000 <sub>H</sub> - A0CF FFFF <sub>H</sub>	1 Mbyte	Program Flash 5 (PF5)	Access	SRIBE
	A0D0 0000 <sub>H</sub> - A0EF FFFF <sub>H</sub>	2 Mbyte	Reserved (for PFLASH)	SRIBE	SRIBE
	A0F0 0000 <sub>H</sub> - A11F FFFF <sub>H</sub>	3 Mbyte	Program Flash 4 (PF4)	Access	SRIBE
	A120 0000 <sub>H</sub> - A1FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Table 28 TC39x Alternate Address Map for SOTA of Segment 10 Erase Counters and Registers**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
10	A800 0000 <sub>H</sub> - A800 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 2 (EC2)	Access	SRIBE
	A800 4000 <sub>H</sub> - A807 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A808 0000 <sub>H</sub> - A80B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 2 (PFI2)	Access	SRIBE
	A80C 0000 <sub>H</sub> - A82F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A830 0000 <sub>H</sub> - A830 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 3 (EC3)	Access	SRIBE
	A830 4000 <sub>H</sub> - A837 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A838 0000 <sub>H</sub> - A83B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 3 (PFI3)	Access	SRIBE
	A83C 0000 <sub>H</sub> - A85F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A860 0000 <sub>H</sub> - A860 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 0 (EC0)	Access	SRIBE
	A860 4000 <sub>H</sub> - A867 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A868 0000 <sub>H</sub> - A86B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 0 (PFI0)	Access	SRIBE
	A86C 0000 <sub>H</sub> - A88F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A890 0000 <sub>H</sub> - A890 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 1 (EC1)	Access	SRIBE
	A890 4000 <sub>H</sub> - A897 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A898 0000 <sub>H</sub> - A89B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 1 (PFI1)	Access	SRIBE
	A89C 0000 <sub>H</sub> - A8BF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A8C0 0000 <sub>H</sub> - A8C0 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 5 (EC5)	Access	SRIBE
	A8C0 4000 <sub>H</sub> - A8C7 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A8C8 0000 <sub>H</sub> - A8CB FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 5 (PFI5)	Access	SRIBE
	A8CC 0000 <sub>H</sub> - A8EF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A8F0 0000 <sub>H</sub> - A8F0 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 4 (EC4)	Access	SRIBE
	A8F0 4000 <sub>H</sub> - A8F7 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A8F8 0000 <sub>H</sub> - A8FB FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 4 (PFI4)	Access	SRIBE
	A8FC 0000 <sub>H</sub> - AEFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Memory Maps (MEMMAP)****Table 29 TC3Ex Alternate Address Map for SOTA of Segment 8 PFLASH**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
8	8000 0000 <sub>H</sub> - 802F FFFF <sub>H</sub>	3 Mbyte	Program Flash 2 (PF2)	Access	SRIBE
	8030 0000 <sub>H</sub> - 805F FFFF <sub>H</sub>	3 Mbyte	Program Flash 3 (PF3)	Access	SRIBE
	8060 0000 <sub>H</sub> - 808F FFFF <sub>H</sub>	3 Mbyte	Program Flash 0 (PF0)	Access	SRIBE
	8090 0000 <sub>H</sub> - 80BF FFFF <sub>H</sub>	3 Mbyte	Program Flash 1 (PF1)	Access	SRIBE
	80C0 0000 <sub>H</sub> - 81FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Table 30 TC3Ex Alternate Address Map for SOTA of Segment 10 PFLASH**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
10	A000 0000 <sub>H</sub> - A02F FFFF <sub>H</sub>	3 Mbyte	Program Flash 2 (PF2)	Access	SRIBE
	A030 0000 <sub>H</sub> - A05F FFFF <sub>H</sub>	3 Mbyte	Program Flash 3 (PF3)	Access	SRIBE
	A060 0000 <sub>H</sub> - A08F FFFF <sub>H</sub>	3 Mbyte	Program Flash 0 (PF0)	Access	SRIBE
	A090 0000 <sub>H</sub> - A0BF FFFF <sub>H</sub>	3 Mbyte	Program Flash 1 (PF1)	Access	SRIBE
	A0C0 0000 <sub>H</sub> - A1FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Table 31 TC3Ex Alternate Address Map for SOTA of Segment 10 Erase Counters and Registers**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
10	A800 0000 <sub>H</sub> - A800 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 2 (EC2)	Access	SRIBE
	A800 4000 <sub>H</sub> - A807 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A808 0000 <sub>H</sub> - A80B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 2 (PFI2)	Access	SRIBE
	A80C 0000 <sub>H</sub> - A82F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A830 0000 <sub>H</sub> - A830 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 3 (EC3)	Access	SRIBE
	A830 4000 <sub>H</sub> - A837 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A838 0000 <sub>H</sub> - A83B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 3 (PFI3)	Access	SRIBE
	A83C 0000 <sub>H</sub> - A85F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A860 0000 <sub>H</sub> - A860 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 0 (EC0)	Access	SRIBE
	A860 4000 <sub>H</sub> - A867 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A868 0000 <sub>H</sub> - A86B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 0 (PFI0)	Access	SRIBE
	A86C 0000 <sub>H</sub> - A88F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A890 0000 <sub>H</sub> - A890 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 1 (EC1)	Access	SRIBE
	A890 4000 <sub>H</sub> - A897 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A898 0000 <sub>H</sub> - A89B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 1 (PFI1)	Access	SRIBE
	A89C 0000 <sub>H</sub> - AEFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Memory Maps (MEMMAP)****Table 32 TC38x Alternate Address Map for SOTA of Segment 8 PFLASH**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
8	8000 0000 <sub>H</sub> - 802F FFFF <sub>H</sub>	3 Mbyte	Program Flash 2 (PF2)	Access	SRIBE
	8030 0000 <sub>H</sub> - 803F FFFF <sub>H</sub>	1 Mbyte	Program Flash 3 (PF3)	Access	SRIBE
	8040 0000 <sub>H</sub> - 805F FFFF <sub>H</sub>	2 Mbyte	Reserved (for PFLASH)	SRIBE	SRIBE
	8060 0000 <sub>H</sub> - 808F FFFF <sub>H</sub>	3 Mbyte	Program Flash 0 (PF0)	Access	SRIBE
	8090 0000 <sub>H</sub> - 80BF FFFF <sub>H</sub>	3 Mbyte	Program Flash 1 (PF1)	Access	SRIBE
	80C0 0000 <sub>H</sub> - 81FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Table 33 TC38x Alternate Address Map for SOTA of Segment 10 PFLASH**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
10	A000 0000 <sub>H</sub> - A02F FFFF <sub>H</sub>	3 Mbyte	Program Flash 2 (PF2)	Access	SRIBE
	A030 0000 <sub>H</sub> - A03F FFFF <sub>H</sub>	1 Mbyte	Program Flash 3 (PF3)	Access	SRIBE
	A040 0000 <sub>H</sub> - A05F FFFF <sub>H</sub>	2 Mbyte	Reserved (for PFLASH)	SRIBE	SRIBE
	A060 0000 <sub>H</sub> - A08F FFFF <sub>H</sub>	3 Mbyte	Program Flash 0 (PF0)	Access	SRIBE
	A090 0000 <sub>H</sub> - A0BF FFFF <sub>H</sub>	3 Mbyte	Program Flash 1 (PF1)	Access	SRIBE
	A0C0 0000 <sub>H</sub> - A1FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Table 34 TC38x Alternate Address Map for SOTA of Segment 10 Erase Counters and Registers**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
10	A800 0000 <sub>H</sub> - A800 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 2 (EC2)	Access	SRIBE
	A800 4000 <sub>H</sub> - A807 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A808 0000 <sub>H</sub> - A80B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 2 (PFI2)	Access	SRIBE
	A80C 0000 <sub>H</sub> - A82F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A830 0000 <sub>H</sub> - A830 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 3 (EC3)	Access	SRIBE
	A830 4000 <sub>H</sub> - A837 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A838 0000 <sub>H</sub> - A83B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 3 (PFI3)	Access	SRIBE
	A83C 0000 <sub>H</sub> - A85F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A860 0000 <sub>H</sub> - A860 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 0 (EC0)	Access	SRIBE
	A860 4000 <sub>H</sub> - A867 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A868 0000 <sub>H</sub> - A86B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 0 (PFI0)	Access	SRIBE
	A86C 0000 <sub>H</sub> - A88F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A890 0000 <sub>H</sub> - A890 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 1 (EC1)	Access	SRIBE
	A890 4000 <sub>H</sub> - A897 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A898 0000 <sub>H</sub> - A89B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 1 (PFI1)	Access	SRIBE
	A89C 0000 <sub>H</sub> - AEFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Memory Maps (MEMMAP)****Table 35 TC37x Alternate Address Map for SOTA of Segment 8 PFLASH**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
8	8000 0000 <sub>H</sub> - 802F FFFF <sub>H</sub>	3 Mbyte	Program Flash 1 (PF1)	Access	SRIBE
	8030 0000 <sub>H</sub> - 805F FFFF <sub>H</sub>	3 Mbyte	Program Flash 0 (PF0)	Access	SRIBE
	8060 0000 <sub>H</sub> - 81FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Table 36 TC37x Alternate Address Map for SOTA of Segment 10 PFLASH**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
10	A000 0000 <sub>H</sub> - A02F FFFF <sub>H</sub>	3 Mbyte	Program Flash 1 (PF1)	Access	SRIBE
	A030 0000 <sub>H</sub> - A05F FFFF <sub>H</sub>	3 Mbyte	Program Flash 0 (PF0)	Access	SRIBE
	A060 0000 <sub>H</sub> - A1FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Table 37 TC37x Alternate Address Map for SOTA of Segment 10 Erase Counters and Registers**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
10	A800 0000 <sub>H</sub> - A800 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 1 (EC1)	Access	SRIBE
	A800 4000 <sub>H</sub> - A807 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A808 0000 <sub>H</sub> - A80B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 1 (PFI1)	Access	SRIBE
	A80C 0000 <sub>H</sub> - A82F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A830 0000 <sub>H</sub> - A830 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 0 (EC0)	Access	SRIBE
	A830 4000 <sub>H</sub> - A837 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A838 0000 <sub>H</sub> - A83B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 0 (PFI0)	Access	SRIBE
	A83C 0000 <sub>H</sub> - AEFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Table 38 TC36x and TC35x Alternate Address Map for SOTA of Segment 8 PFLASH**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
8	8000 0000 <sub>H</sub> - 801F FFFF <sub>H</sub>	2 Mbyte	Program Flash 1 (PF1)	Access	SRIBE
	8020 0000 <sub>H</sub> - 802F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	8030 0000 <sub>H</sub> - 804F FFFF <sub>H</sub>	2 Mbyte	Program Flash 0 (PF0)	Access	SRIBE
	8050 0000 <sub>H</sub> - 81FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Table 39 TC36x and TC35x Alternate Address Map for SOTA of Segment 10 PFLASH**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
10	A000 0000 <sub>H</sub> - A01F FFFF <sub>H</sub>	2 Mbyte	Program Flash 1 (PF1)	Access	SRIBE
	A020 0000 <sub>H</sub> - A02F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

## Memory Maps (MEMMAP)

**Table 39 TC36x and TC35x Alternate Address Map for SOTA (cont'd) of Segment 10 PFLASH**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
	A030 0000 <sub>H</sub> - A04F FFFF <sub>H</sub>	2 Mbyte	Program Flash 0 (PF0)	Access	SRIBE
	A050 0000 <sub>H</sub> - A1FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

**Table 40 TC36x and TC35x Alternate Address Map for SOTA of Segment 10 Erase Counters and Registers**

Segment	Address Range	Size	Description	Access Type	
				Read	Write
10	A800 0000 <sub>H</sub> - A800 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 1 (EC1)	Access	SRIBE
	A800 4000 <sub>H</sub> - A807 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A808 0000 <sub>H</sub> - A80B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 1 (PFI1)	Access	SRIBE
	A80C 0000 <sub>H</sub> - A82F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A830 0000 <sub>H</sub> - A830 3FFF <sub>H</sub>	16 Kbyte	Erase Counter 0 (EC0)	Access	SRIBE
	A830 4000 <sub>H</sub> - A837 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
	A838 0000 <sub>H</sub> - A83B FFFF <sub>H</sub>	256 Kbyte	PFI User Registers 0 (PFI0)	Access	SRIBE
	A83C 0000 <sub>H</sub> - AEFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

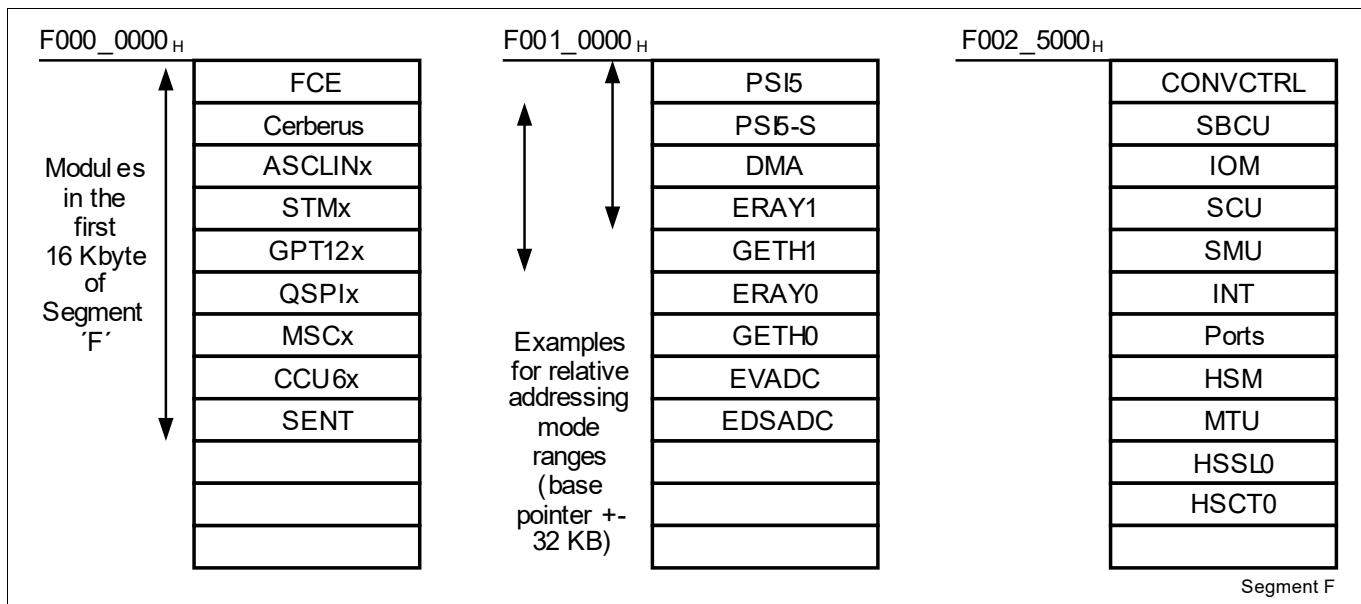
### 2.3.2.2 Segment 15

**Table 41** shows the address map of segment 'F' as seen from the SRI and SPB bus masters (bus master agents are described in the chapter On Chip Bus System). It describes the mapping of modules to Segment F. The details of the module address ranges can be found in the module chapters register overview.

**Figure 17** gives an overview about the address mapping of the module address ranges:

- Absolute Addressing Range
  - If a module is addressed in the first 16 Kbyte of segment 'F', the CPU can access the module with absolute addressing mode.
- Others
  - If a module is addressed above the first 16 Kbyte of segment 'F', the CPU can access the module with base + offset.

## Memory Maps (MEMMAP)



**Figure 17 Segment F Structure**

**Table 41 Address Map of Segment 15**

Address Range	Size	Unit	Access Type	
			Read	Write
F000 0000 <sub>H</sub> - F000 01FF <sub>H</sub>	512 Byte	Flexible CRC Engine (FCE0)	Access	Access
F000 0200 <sub>H</sub> - F000 03FF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F000 0400 <sub>H</sub> - F000 05FF <sub>H</sub>	2 x 256 Byte	On-Chip Debug Support (Cerberus)	Access	Access
F000 0600 <sub>H</sub> - F000 06FF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 0 (ASCLIN0)	Access	Access
F000 0700 <sub>H</sub> - F000 07FF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 1 (ASCLIN1)	Access	Access
F000 0800 <sub>H</sub> - F000 08FF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 2 (ASCLIN2)	Access	Access
F000 0900 <sub>H</sub> - F000 09FF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 3 (ASCLIN3)	Access	Access
F000 0A00 <sub>H</sub> - F000 0AFF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 4 (ASCLIN4)	Access	Access
F000 0B00 <sub>H</sub> - F000 0BFF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 5 (ASCLIN5)	Access	Access
F000 0C00 <sub>H</sub> - F000 0CFF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 6 (ASCLIN6)	Access	Access
F000 0D00 <sub>H</sub> - F000 0DFF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 7 (ASCLIN7)	Access	Access
F000 0E00 <sub>H</sub> - F000 0EFF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 8 (ASCLIN8)	Access	Access
F000 0F00 <sub>H</sub> - F000 0FFF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 9 (ASCLIN9)	Access	Access

**Memory Maps (MEMMAP)****Table 41 Address Map of Segment 15 (cont'd)**

<b>Address Range</b>	<b>Size</b>	<b>Unit</b>	<b>Access Type</b>	
			<b>Read</b>	<b>Write</b>
F000 1000 <sub>H</sub> - F000 10FF <sub>H</sub>	256 Byte	System Timer 0 (STM0)	Access	Access
F000 1100 <sub>H</sub> - F000 11FF <sub>H</sub>	256 Byte	System Timer 1 (STM1)	Access	Access
F000 1200 <sub>H</sub> - F000 12FF <sub>H</sub>	256 Byte	System Timer 2 (STM2)	Access	Access
F000 1300 <sub>H</sub> - F000 13FF <sub>H</sub>	256 Byte	System Timer 3 (STM3)	Access	Access
F000 1400 <sub>H</sub> - F000 14FF <sub>H</sub>	256 Byte	System Timer 4 (STM4)	Access	Access
F000 1500 <sub>H</sub> - F000 15FF <sub>H</sub>	256 Byte	System Timer 5 (STM5)	Access	Access
F000 1600 <sub>H</sub> - F000 17FF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F000 1800 <sub>H</sub> - F000 18FF <sub>H</sub>	256 Byte	General Purpose Timer Unit (GPT120)	Access	Access
F000 1900 <sub>H</sub> - F000 1BFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F000 1C00 <sub>H</sub> - F000 1CFF <sub>H</sub>	256 Byte	Queued SPI Controller 0 (QSPI0)	Access	Access
F000 1D00 <sub>H</sub> - F000 1DFF <sub>H</sub>	256 Byte	Queued SPI Controller 1 (QSPI1)	Access	Access
F000 1E00 <sub>H</sub> - F000 1EFF <sub>H</sub>	256 Byte	Queued SPI Controller 2 (QSPI2)	Access	Access
F000 1F00 <sub>H</sub> - F000 1FFF <sub>H</sub>	256 Byte	Queued SPI Controller 3 (QSPI3)	Access	Access
F000 2000 <sub>H</sub> - F000 20FF <sub>H</sub>	256 Byte	Queued SPI Controller 4 (QSPI4)	Access	Access
F000 2100 <sub>H</sub> - F000 21FF <sub>H</sub>	256 Byte	Queued SPI Controller 5 (QSPI5)	Access	Access
F000 2200 <sub>H</sub> - F000 25FF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F000 2600 <sub>H</sub> - F000 26FF <sub>H</sub>	256 Byte	MicroSecond Bus Controller 0 (MSC0)	Access	Access
F000 2700 <sub>H</sub> - F000 27FF <sub>H</sub>	256 Byte	MicroSecond Bus Controller 1 (MSC1)	Access	Access
F000 2800 <sub>H</sub> - F000 28FF <sub>H</sub>	256 Byte	MicroSecond Bus Controller 2 (MSC2)	Access	Access
F000 2900 <sub>H</sub> - F000 29FF <sub>H</sub>	256 Byte	MicroSecond Bus Controller 3 (MSC3)	Access	Access
F000 2A00 <sub>H</sub> - F000 2AFF <sub>H</sub>	256 Byte	Capture/Compare Unit 6 0 (CCU60)	Access	Access
F000 2B00 <sub>H</sub> - F000 2BFF <sub>H</sub>	256 Byte	Capture/Compare Unit 6 1 (CCU61)	Access	Access
F000 2C00 <sub>H</sub> - F000 2FFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F000 3000 <sub>H</sub> - F000 3AFF <sub>H</sub>	11 x 256 Byte	Single Edge Nibble Transmission (SENT)	Access	Access
F000 3B00 <sub>H</sub> - F000 4FFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F000 5000 <sub>H</sub> - F000 5AFF <sub>H</sub>	11 x 256 Byte	Peripheral Sensor Interface (PSI5)	Access	Access
F000 5B00 <sub>H</sub> - F000 6FFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F000 7000 <sub>H</sub> - F000 7FFF <sub>H</sub>	4 Kbyte	Peripheral Sensor Interface-S (PSI5S)	Access	Access
F000 8000 <sub>H</sub> - F000 FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F001 0000 <sub>H</sub> - F001 3FFF <sub>H</sub>	16 Kbyte	Direct Memory Access Controller (DMA)	Access	Access
F001 4000 <sub>H</sub> - F001 6FFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F001 7000 <sub>H</sub> - F001 7FFF <sub>H</sub>	4 Kbyte	FlexRay™ Protocol Controller 1 (ERAY1)	Access	Access
F001 9000 <sub>H</sub> - F001 9FFF <sub>H</sub>	4 Kbyte	Gigabit Ethernet Controller MAC Control (GETH1)	Access	Access
F001 A000 <sub>H</sub> - F001 AFFF <sub>H</sub>	4 Kbyte	Gigabit Ethernet Controller DMA Control (GETH1)	Access	Access
F001 B000 <sub>H</sub> - F001 B0FF <sub>H</sub>	256 Byte	Gigabit Ethernet Controller SFR (GETH1)	Access	Access

**Memory Maps (MEMMAP)****Table 41 Address Map of Segment 15 (cont'd)**

<b>Address Range</b>	<b>Size</b>	<b>Unit</b>	<b>Access Type</b>	
			<b>Read</b>	<b>Write</b>
F001 B100 <sub>H</sub> - F001 BFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F001 C000 <sub>H</sub> - F001 CFFF <sub>H</sub>	4 Kbyte	FlexRayTM Protocol Controller 0 (ERAY0)	Access	Access
F001 D000 <sub>H</sub> - F001 DFFF <sub>H</sub>	4 Kbyte	Gigabit Ethernet Controller MAC Control (GETH0)	Access	Access
F001 E000 <sub>H</sub> - F001 EFFF <sub>H</sub>	4 Kbyte	Gigabit Ethernet Controller DMA Control (GETH0)	Access	Access
F001 F000 <sub>H</sub> - F001 F0FF <sub>H</sub>	256 Byte	Gigabit Ethernet Controller SFR (GETH0)	Access	Access
F001 F100 <sub>H</sub> - F001 FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F002 0000 <sub>H</sub> - F002 3FFF <sub>H</sub>	16 Kbyte	Analog-to-Digital Converter (EVADC)	Access	Access
F002 4000 <sub>H</sub> - F002 4FFF <sub>H</sub>	4 Kbyte	Delta Sigma Analog-to-Digital Converter (EDSADC)	Access	Access
F002 5000 <sub>H</sub> - F002 50FF <sub>H</sub>	256 Byte	Converter Control (CONVCTRL)	Access	Access
F002 5100 <sub>H</sub> - F002 FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F003 0000 <sub>H</sub> - F003 00FF <sub>H</sub>	256 Byte	SPB Bus Control Unit (SBCU)	Access	Access
F003 0100 <sub>H</sub> - F003 4FFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F003 5000 <sub>H</sub> - F003 51FF <sub>H</sub>	2 x 256 Byte	I/O Monitor (IOM)	Access	Access
F003 5200 <sub>H</sub> - F003 5FFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F003 6000 <sub>H</sub> - F003 63FF <sub>H</sub>	1 Kbyte	System Control Unit (SCU)	Access	Access
F003 6400 <sub>H</sub> - F003 67FF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F003 6800 <sub>H</sub> - F003 6FFF <sub>H</sub>	2 Kbyte	Safety Management Unit (SMU)	Access	Access
F003 7000 <sub>H</sub> - F003 7FFF <sub>H</sub>	4 Kbyte	Interrupt Router (INT)	Access	Access
F003 8000 <sub>H</sub> - F003 9FFF <sub>H</sub>	8 Kbyte	Interrupt Router SRC Registers (INT)	Access	Access
F003 A000 <sub>H</sub> - F003 A0FF <sub>H</sub>	256 Byte	Port 00 (P00)	Access	Access
F003 A100 <sub>H</sub> - F003 A1FF <sub>H</sub>	256 Byte	Port 01 (P01)	Access	Access
F003 A200 <sub>H</sub> - F003 A2FF <sub>H</sub>	256 Byte	Port 02 (P02)	Access	Access
F003 A300 <sub>H</sub> - F003 A9FF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F003 AA00 <sub>H</sub> - F003 AAFF <sub>H</sub>	256 Byte	Port 10 (P10)	Access	Access
F003 AB00 <sub>H</sub> - F003 ABFF <sub>H</sub>	256 Byte	Port 11 (P11)	Access	Access
F003 AC00 <sub>H</sub> - F003 ACFF <sub>H</sub>	256 Byte	Port 12 (P12)	Access	Access
F003 AD00 <sub>H</sub> - F003 ADFF <sub>H</sub>	256 Byte	Port 13 (P13)	Access	Access
F003 AE00 <sub>H</sub> - F003 AEFF <sub>H</sub>	256 Byte	Port 14 (P14)	Access	Access
F003 AF00 <sub>H</sub> - F003 AFFF <sub>H</sub>	256 Byte	Port 15 (P15)	Access	Access
F003 B000 <sub>H</sub> - F003 B3FF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F003 B400 <sub>H</sub> - F003 B4FF <sub>H</sub>	256 Byte	Port 20 (P20)	Access	Access
F003 B500 <sub>H</sub> - F003 B5FF <sub>H</sub>	256 Byte	Port 21 (P21)	Access	Access
F003 B600 <sub>H</sub> - F003 B6FF <sub>H</sub>	256 Byte	Port 22 (P22)	Access	Access
F003 B700 <sub>H</sub> - F003 B7FF <sub>H</sub>	256 Byte	Port 23 (P23)	Access	Access

**Memory Maps (MEMMAP)****Table 41 Address Map of Segment 15 (cont'd)**

<b>Address Range</b>	<b>Size</b>	<b>Unit</b>	<b>Access Type</b>	
			<b>Read</b>	<b>Write</b>
F003 B800 <sub>H</sub> - F003 B8FF <sub>H</sub>	256 Byte	Port 24 (P24)	Access	Access
F003 B900 <sub>H</sub> - F003 B9FF <sub>H</sub>	256 Byte	Port 25 (P25)	Access	Access
F003 BA00 <sub>H</sub> - F003 BAFF <sub>H</sub>	256 Byte	Port 26 (P26)	Access	Access
F003 BB00 <sub>H</sub> - F003 BDFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F003 BE00 <sub>H</sub> - F003 BEFF <sub>H</sub>	256 Byte	Port 30 (P30)	Access	Access
F003 BF00 <sub>H</sub> - F003 BFFF <sub>H</sub>	256 Byte	Port 31 (P31)	Access	Access
F003 C000 <sub>H</sub> - F003 COFF <sub>H</sub>	256 Byte	Port 32 (P32)	Access	Access
F003 C100 <sub>H</sub> - F003 C1FF <sub>H</sub>	256 Byte	Port 33 (P33)	Access	Access
F003 C200 <sub>H</sub> - F003 C2FF <sub>H</sub>	256 Byte	Port 34 (P34)	Access	Access
F003 C300 <sub>H</sub> - F003 C7FF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F003 C800 <sub>H</sub> - F003 C8FF <sub>H</sub>	256 Byte	Port 40 (P40)	Access	Access
F003 C900 <sub>H</sub> - F003 C9FF <sub>H</sub>	256 Byte	Port 41 (P41)	Access	Access
F003 CA00 <sub>H</sub> - F003 FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F004 0000 <sub>H</sub> - F005 FFFF <sub>H</sub>	128 Kbyte	Hardware Security Module (HSM)	Access	Access
F006 0000 <sub>H</sub> - F006 FFFF <sub>H</sub>	64 Kbyte	Memory Test Unit (MTU)	Access	Access
F007 0000 <sub>H</sub> - F007 FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F008 0000 <sub>H</sub> - F008 03FF <sub>H</sub>	4 x 256 Byte	High Speed Serial Link (HSSL0)	Access	Access
F008 0400 <sub>H</sub> - F008 FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F009 0000 <sub>H</sub> - F009 FFFF <sub>H</sub>	64 Kbyte	High Speed Communication Tunnel (HSCT0)	Access	Access
F00A 0000 <sub>H</sub> - F00A 03FF <sub>H</sub>	4 x 256 Byte	High Speed Serial Link (HSSL1)	Access	Access
F00A 0400 <sub>H</sub> - F00A FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F00B 0000 <sub>H</sub> - F00B FFFF <sub>H</sub>	64 Kbyte	High Speed Communication Tunnel (HSCT1)	Access	Access
F00C 0000 <sub>H</sub> - F00C FFFF <sub>H</sub>	64 Kbyte	I2C0 (I2C0)	Access	Access
F00D 0000 <sub>H</sub> - F00D 00FF <sub>H</sub>	256 Byte	I2C0 System Control Register (I2C0)	Access	Access
F00D 0100 <sub>H</sub> - F00D FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F00E 0000 <sub>H</sub> - F00E FFFF <sub>H</sub>	64 Kbyte	I2C1 (I2C1)	Access	Access
F00F 0000 <sub>H</sub> - F00F 00FF <sub>H</sub>	256 Byte	I2C1 System Control Register (I2C1)	Access	Access
F00F 0100 <sub>H</sub> - F00F FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F010 0000 <sub>H</sub> - F01F FFFF <sub>H</sub>	1 Mbyte	Generic Timer Module (GTM)	Access	Access
F020 0000 <sub>H</sub> - F020 7FFF <sub>H</sub>	32 Kbyte	MCMCAN0 SRAM (CAN0)	Access	Access
F020 8000 <sub>H</sub> - F020 8FFF <sub>H</sub>	4 Kbyte	MCMCAN0 SFR (CAN0)	Access	Access
F020 9000 <sub>H</sub> - F020 FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F021 0000 <sub>H</sub> - F021 3FFF <sub>H</sub>	16 Kbyte	MCMCAN1 SRAM (CAN1)	Access	Access
F021 4000 <sub>H</sub> - F021 7FFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F021 8000 <sub>H</sub> - F021 8FFF <sub>H</sub>	4 Kbyte	MCMCAN1 SFR (CAN1)	Access	Access

**Memory Maps (MEMMAP)****Table 41 Address Map of Segment 15 (cont'd)**

<b>Address Range</b>	<b>Size</b>	<b>Unit</b>	<b>Access Type</b>	
			<b>Read</b>	<b>Write</b>
F021 9000 <sub>H</sub> - F021 FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F022 0000 <sub>H</sub> - F022 3FFF <sub>H</sub>	16 Kbyte	MCMCAN2 SRAM (CAN2)	Access	Access
F022 4000 <sub>H</sub> - F022 7FFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F022 8000 <sub>H</sub> - F022 8FFF <sub>H</sub>	4 Kbyte	MCMCAN2 SFR (CAN2)	Access	Access
F022 9000 <sub>H</sub> - F022 FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F023 0000 <sub>H</sub> - F023 3FFF <sub>H</sub>	16 Kbyte	MCMCAN3 SRAM (CAN3)	Access	Access
F023 4000 <sub>H</sub> - F023 7FFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F023 8000 <sub>H</sub> - F023 8FFF <sub>H</sub>	4 Kbyte	MCMCAN3 SFR (CAN3)	Access	Access
F023 9000 <sub>H</sub> - F023 FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F024 0000 <sub>H</sub> - F024 1FFF <sub>H</sub>	8 Kbyte	Standby Controller XRAM (SCR XRAM)	Access	Access
F024 2000 <sub>H</sub> - F024 7FFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F024 8000 <sub>H</sub> - F024 81FF <sub>H</sub>	512 Byte	Power Management System (PMS)	Access	Access
F024 8200 <sub>H</sub> - F027 FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F028 0000 <sub>H</sub> - F028 1FFF <sub>H</sub>	8 Kbyte	High Speed Pulse Density Modulation SRAM (HSPDM)	SPBBE	SPBBE
F028 2000 <sub>H</sub> - F028 20FF <sub>H</sub>	256 Byte	High Speed Pulse Density Modulation SFR (HSPDM)	SPBBE	SPBBE
F028 2100 <sub>H</sub> - F02A FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F02B 0000 <sub>H</sub> - F02B 0FFF <sub>H</sub>	4 Kbyte	Secure Digital Multi Media Card (SDMMC0)	Access	Access
F02B 0200 <sub>H</sub> - F02C 09FF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F02C 0A00 <sub>H</sub> - F02C 0AFF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 10 (ASCLIN10)	Access	Access
F02C 0B00 <sub>H</sub> - F02C 0BFF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 11 (ASCLIN11)	Access	Access
F02C 0C00 <sub>H</sub> - F02C 0CFF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 12 (ASCLIN12) <sup>1)</sup>	Access	Access
F02C 0D00 <sub>H</sub> - F02C 0DFF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 13 (ASCLIN13) <sup>1)</sup>	Access	Access
F02C 0E00 <sub>H</sub> - F02C 0EFF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 14 (ASCLIN14) <sup>1)</sup>	Access	Access
F02C 0F00 <sub>H</sub> - F02C 0FFF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 15 (ASCLIN15) <sup>1)</sup>	Access	Access
F02C 1000 <sub>H</sub> - F02C 10FF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 16 (ASCLIN16) <sup>1)</sup>	Access	Access
F02C 1100 <sub>H</sub> - F02C 11FF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 17 (ASCLIN17) <sup>1)</sup>	Access	Access
F02C 1200 <sub>H</sub> - F02C 12FF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 18 (ASCLIN18) <sup>1)</sup>	Access	Access

**Memory Maps (MEMMAP)****Table 41 Address Map of Segment 15 (cont'd)**

<b>Address Range</b>	<b>Size</b>	<b>Unit</b>	<b>Access Type</b>	
			<b>Read</b>	<b>Write</b>
F02C 1300 <sub>H</sub> - F02C 13FF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 19 (ASCLIN19) <sup>1)</sup>	Access	Access
F02C 1400 <sub>H</sub> - F02C 14FF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 20 (ASCLIN20) <sup>1)</sup>	Access	Access
F02C 1500 <sub>H</sub> - F02C 15FF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 21 (ASCLIN21) <sup>1)</sup>	Access	Access
F02C 1600 <sub>H</sub> - F02C 16FF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 22 (ASCLIN22) <sup>1)</sup>	Access	Access
F02C 1700 <sub>H</sub> - F02C 17FF <sub>H</sub>	256 Byte	Asynchronous/Synchronous Serial Controller with LIN 23 (ASCLIN23) <sup>1)</sup>	Access	Access
F02C 1800 <sub>H</sub> - F033 FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F034 0000 <sub>H</sub> - F034 3FFF <sub>H</sub>	16 Kbyte	MCMCAN4 SRAM (CAN4)	Access	Access
F034 4000 <sub>H</sub> - F034 7FFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F034 8000 <sub>H</sub> - F034 8FFF <sub>H</sub>	4 Kbyte	MCMCAN4 SFR (CAN4)	Access	Access
F034 9000 <sub>H</sub> - F7FF FFFF <sub>H</sub>	-	Reserved	SPBBE	SPBBE
F800 0000 <sub>H</sub> - F801 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
F802 0000 <sub>H</sub> - F802 FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
F803 0000 <sub>H</sub> - F803 00FF <sub>H</sub>	256 Byte	FSI SFR (DMU)	Access	Access
F803 0100 <sub>H</sub> - F803 7FFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
F803 8000 <sub>H</sub> - F803 FFFF <sub>H</sub>	32 Kbyte	Boot ROM Control (DMU)	Access	Access
F804 0000 <sub>H</sub> - F804 FFFF <sub>H</sub>	64 Kbyte	Host Command Interface (DMU)	Access	Access
F805 0000 <sub>H</sub> - F805 FFFF <sub>H</sub>	64 Kbyte	Host Protection Configuration (DMU)	Access	Access
F806 0000 <sub>H</sub> - F806 FFFF <sub>H</sub>	64 Kbyte	HSM Command Interface (DMU)	Access	Access
F807 0000 <sub>H</sub> - F807 FFFF <sub>H</sub>	64 Kbyte	HSM Protection Configuration (DMU)	Access	Access
F808 0000 <sub>H</sub> - F80F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
F810 0000 <sub>H</sub> - F810 FFFF <sub>H</sub>	64 Kbyte	Local Memory Unit (LMU0)	Access	Access
F811 0000 <sub>H</sub> - F811 FFFF <sub>H</sub>	64 Kbyte	Local Memory Unit (LMU1)	Access	Access
F812 0000 <sub>H</sub> - F812 FFFF <sub>H</sub>	64 Kbyte	Local Memory Unit (LMU2)	Access	Access
F813 0000 <sub>H</sub> - F83F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
F840 0000 <sub>H</sub> - F840 FFFF <sub>H</sub>	64 Kbyte	External Bus Unit (EBU0)	Access	Access
F841 0000 <sub>H</sub> - F84F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
F850 0000 <sub>H</sub> - F850 FFFF <sub>H</sub>	64 Kbyte	DAM (DAM0)	Access	Access
F851 0000 <sub>H</sub> - F851 FFFF <sub>H</sub>	64 Kbyte	DAM (DAM1)	Access	Access
F852 0000 <sub>H</sub> - F86F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
F870 0000 <sub>H</sub> - F870 FFFF <sub>H</sub>	64 Kbyte	SRI Domain 0 SFR (SRI0)	Access	Access
F871 0000 <sub>H</sub> - F87F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
F880 0000 <sub>H</sub> - F880 FFFF <sub>H</sub>	64 Kbyte	CPU0 SFR (CPU0)	Access	Access
F881 0000 <sub>H</sub> - F881 FFFF <sub>H</sub>	64 Kbyte	CPU0 CSFR (CPU0)	Access	Access

**Memory Maps (MEMMAP)****Table 41 Address Map of Segment 15 (cont'd)**

<b>Address Range</b>	<b>Size</b>	<b>Unit</b>	<b>Access Type</b>	
			<b>Read</b>	<b>Write</b>
F882 0000 <sub>H</sub> - F882 FFFF <sub>H</sub>	64 Kbyte	CPU1 SFR (CPU1)	Access	Access
F883 0000 <sub>H</sub> - F883 FFFF <sub>H</sub>	64 Kbyte	CPU1 CSFR (CPU1)	Access	Access
F884 0000 <sub>H</sub> - F884 FFFF <sub>H</sub>	64 Kbyte	CPU2 SFR (CPU2)	Access	Access
F885 0000 <sub>H</sub> - F885 FFFF <sub>H</sub>	64 Kbyte	CPU2 CSFR (CPU2)	Access	Access
F886 0000 <sub>H</sub> - F886 FFFF <sub>H</sub>	64 Kbyte	CPU3 SFR (CPU3)	Access	Access
F887 0000 <sub>H</sub> - F887 FFFF <sub>H</sub>	64 Kbyte	CPU3 CSFR (CPU3)	Access	Access
F888 0000 <sub>H</sub> - F888 FFFF <sub>H</sub>	64 Kbyte	CPU4 SFR (CPU4)	Access	Access
F889 0000 <sub>H</sub> - F889 FFFF <sub>H</sub>	64 Kbyte	CPU4 CSFR (CPU4)	Access	Access
F88A 0000 <sub>H</sub> - F88B FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
F88C 0000 <sub>H</sub> - F88C FFFF <sub>H</sub>	64 Kbyte	CPU5 SFR (CPU5)	Access	Access
F88D 0000 <sub>H</sub> - F88D FFFF <sub>H</sub>	64 Kbyte	CPU5 CSFR (CPU5)	Access	Access
F88E 0000 <sub>H</sub> - F88E FFFF <sub>H</sub>	64 Kbyte	SRI Domain 1 SFR (SRI1)	Access	Access
F88F 0000 <sub>H</sub> - F9FF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
FA00 0000 <sub>H</sub> - FA00 00FF <sub>H</sub>	-	Reserved	BBBBE	BBBBE
FA00 0100 <sub>H</sub> - FA00 01FF <sub>H</sub>	256 Byte	BBB Bus Control Unit (EBCU)	Access	Access
FA00 0200 <sub>H</sub> - FA00 0FFF <sub>H</sub>	-	Reserved	BBBBE	BBBBE
FA00 1000 <sub>H</sub> - FA00 10FF <sub>H</sub>	256 Byte	AGBT	Access	Access
FA00 1100 <sub>H</sub> - FA00 5EFF <sub>H</sub>	-	Reserved	BBBBE	BBBBE
FA00 6000 <sub>H</sub> - FA00 60FF <sub>H</sub>	256 Byte	EMEM Control Registers	Access	Access
FA00 6100 <sub>H</sub> - FA00 FFFF <sub>H</sub>	-	Reserved	BBBBE	BBBBE
FA01 0000 <sub>H</sub> - FA01 FFFF <sub>H</sub>	64 Kbyte	MCDS	Access	Access
FA02 0000 <sub>H</sub> - FA03 FFFF <sub>H</sub>	-	Reserved	BBBBE	BBBBE
FA04 0000 <sub>H</sub> - FA04 01FF <sub>H</sub>	512 Byte	Radar Interface 0 SFR (RIFO)	Access	Access
FA04 0200 <sub>H</sub> - FA04 03FF <sub>H</sub>	512 Byte	Radar Interface 1 SFR (RIF1)	Access	Access
FA04 0400 <sub>H</sub> - FA6F FFFF <sub>H</sub>	-	Reserved	BBBBE	BBBBE
FA70 0000 <sub>H</sub> - FA70 00FF <sub>H</sub>	256 Byte	SPU Lockstep SFR	Access	Access
FA70 0100 <sub>H</sub> - FA70 01FF <sub>H</sub>	256 Byte	BITMGR	Access	Access
FA70 0200 <sub>H</sub> - FA7F FFFF <sub>H</sub>	-	Reserved	BBBBE	BBBBE
FA80 0000 <sub>H</sub> - FA80 07FF <sub>H</sub>	2 Kbyte	Signal Processing Unit 0 SFR (SPU0)	Access	Access
FA80 0800 <sub>H</sub> - FA9F FFFF <sub>H</sub>	-	Reserved	BBBBE	BBBBE
FAA0 0000 <sub>H</sub> - FAA1 FFFF <sub>H</sub>	128 Kbyte	SPU0 Configuration RAM (SPUCFG0)	Access	Access
FAA2 0000 <sub>H</sub> - FABF FFFF <sub>H</sub>	-	Reserved	BBBBE	BBBBE
FAC0 0000 <sub>H</sub> - FAC0 07FF <sub>H</sub>	2 Kbyte	Signal Processing Unit 1 SFR (SPU1)	Access	Access
FAC0 0800 <sub>H</sub> - FADF FFFF <sub>H</sub>	-	Reserved	BBBBE	BBBBE
FAE0 0000 <sub>H</sub> - FAE1 FFFF <sub>H</sub>	128 Kbyte	SPU1 Configuration RAM (SPUCFG1)	Access	Access
FAE2 0000 <sub>H</sub> - FAFF FFFF <sub>H</sub>	-	Reserved	BBBBE	BBBBE
FB00 0000 <sub>H</sub> - FB00 FFFF <sub>H</sub>	64 Kbyte	EMEM SRI Slave Interface 0 SFR	Access	Access

## Memory Maps (MEMMAP)

**Table 41 Address Map of Segment 15 (cont'd)**

Address Range		Size	Unit	Access Type	
				Read	Write
FB01 0000 <sub>H</sub>	- FB01 FFFF <sub>H</sub>	64 Kbyte	EMEM SRI Slave Interface 1 SFR	Access	Access
FB02 0000 <sub>H</sub>	- FB02 FFFF <sub>H</sub>	64 Kbyte	EMEM SRI Slave Interface 2 SFR	Access	Access
FB03 0000 <sub>H</sub>	- FB03 FFFF <sub>H</sub>	64 Kbyte	EMEM SRI Slave Interface 3 SFR	Access	Access
FB04 0000 <sub>H</sub>	- FB04 FFFF <sub>H</sub>	64 Kbyte	EMEM SRI Slave Interface 4 SFR	Access	Access
FB05 0000 <sub>H</sub>	- FB05 FFFF <sub>H</sub>	64 Kbyte	EMEM SRI Slave Interface 5 SFR	Access	Access
FB06 0000 <sub>H</sub>	- FB6F FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
FB70 0000 <sub>H</sub>	- FB70 FFFF <sub>H</sub>	64 Kbyte	SRI Domain 2 SFR (SRI2)	Access	Access
FB71 0000 <sub>H</sub>	- FB71 7FFF <sub>H</sub>	32 Kbyte	Reserved	SRIBE	SRIBE
FB71 8000 <sub>H</sub>	- FB71 FFFF <sub>H</sub>	32 Kbyte	MINIMCDS SFR (MINIMCDS)	Access <sup>2)</sup>	Access <sup>2)</sup>
FB72 0000 <sub>H</sub>	- FBFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
FC00 0000 <sub>H</sub>	- FFBF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE
FFC0 0000 <sub>H</sub>	- FFC1 FFFF <sub>H</sub>	128 Kbyte	Data Flash 1 EEPROM (DF1) HSM Command Sequence Interpreter	Access	Access <sup>3)</sup>
FFC2 0000 <sub>H</sub>	- FFC3 FFFF <sub>H</sub>	128 Kbyte	Reserved	SRIBE	SRIBE
FFC4 0000 <sub>H</sub>	- FFFF FFFF <sub>H</sub>	-	Reserved	SRIBE	SRIBE

1) TC38x only

2) MINIMCDS SFR may only be accessed when OCDS is enabled.

3) HSM Command Sequence Interpreter

### 2.3.3 Memory Accesses

The following tables list the types of memories and supported access sizes<sup>1)</sup>:

**Table 42 Standard Read Write Memories (C variable)**

Memory	Byte		Half-word		Word			Double-word		Block Transfer	
	r	w	r	w	r	w	rmw	r	w	r	w
PSPR	y	y	y	y	y	y	y	y	y	y	y
DSPR	y	y	y	y	y	y	y	y	y	y	y
DLMU	y	y	y	y	y	y	y	y	y	y	y
LMURAM	y	y	y	y	y	y	y	y	y	y	y
EMEM	y	y	y	y	y	y	y	y	y	y	y
DAM RAM	y	y	y	y	y	y	y	y	y	y	y
TRAM <sup>1)</sup>	y	y	y	y	y	y	y	y	y	y	y

1) TRAM shall not be used as a general SRAM and can only be accessed when OCDS is enabled.

1) 'y' means: access supported. '-' means: access not supported.

## Memory Maps (MEMMAP)

**Table 43 Standard Read Only Memories (C const)**

Memory	Byte		Half-word		Word			Double-word		Block Transfer	
	r	w	r	w	r	w	rmw	r	w	r	w
BROM	y	-	y	-	y	-	-	y	-	-	-
PFLASH <sup>1)</sup>	y	-	y	-	y	-	-	y	-	y	-
DFLASH <sup>1)</sup>	y	-	y	-	y	-	-	y	-	-	-

1) FLASH memory shall be programmed and erased by command sequences.

**Table 44 Non Standard Memories**

Memory	Byte		Half-word		Word			Double-word		Block Transfer	
	r	w	r	w	r	w	rmw	r	w	r	w
PTAG <sup>1)</sup>	-	-	-	-	y	y	-	-	-	-	-
PCACHE	y	y	y	y	y	y	y	y	y	y	y
DTAG <sup>1)</sup>	-	-	-	-	y	y	-	-	-	-	-
DCACHE	y	y	y	y	y	y	y	y	y	y	y

1) TAG SRAMs are not meant to be used as general SRAMs and can be accessed only with 32-bit data access and only with 64 bit aligned address. Mapping of TAG SRAMs in the address map can be used as additional option for memory testing. The TAG SRAM size is below 24 bit, so the 8 MSB of a 32 bit write or read are don't care.

## 2.4 Revision History

**Table 45 Changes**

Reference	Change to Previous Version	Comment
<b>V0.1.8</b>	- First-level heading structure adjusted.	
<b>V0.1.9</b>		
<a href="#">Table 35</a>	TC37x Alternate Address Map for SOTA of Segment 8 PFLASH	
<a href="#">Table 36</a>	TC37x Alternate Address Map for SOTA of Segment 10 PFLASH	
<a href="#">Table 37</a>	TC37x Alternate Address Map for SOTA of Segment 10 Erase Counters and Registers	
<a href="#">Table 38</a>	TC36x and TC3SPUCFG15x Alternate Address Map for SOTA of Segment 8 PFLASH	
<a href="#">Table 39</a>	TC36x and TC35x Alternate Address Map for SOTA of Segment 10 PFLASH	
<a href="#">Table 40</a>	TC36x and TC35x Alternate Address Map for SOTA of Segment 10 Erase Counters and Registers	
<b>V0.1.10</b>		
<a href="#">Table 41</a>	Add Gigabit Ethernet Controller (GETH1) from F001 8000H to F001 A0FFH	
<b>V0.1.11</b>		
<a href="#">Page 1</a>	Add sentence for device specific MEMMAP.	
<a href="#">Page 1</a>	For CPU change "LMU" to "DLMU"	
<a href="#">Page 15</a>	GETH1 address range shifted.	

**Memory Maps (MEMMAP)****Table 45 Changes**

Reference	Change to Previous Version	Comment
<b>V0.1.12</b>		
<a href="#">Page 15</a>	Add MCMCAN3.	
<b>V0.1.13</b>		
<a href="#">Page 15</a>	Add MCMCAN4.	
<b>V0.1.14</b>		
<a href="#">Page 12</a>	TC38x Alternate Address Map for SOTA of Segment 8 PFLASH	
<a href="#">Page 12</a>	TC38x Alternate Address Map for SOTA of Segment 10 PFLASH	
<a href="#">Page 12</a>	TC38x Alternate Address Map for SOTA of Segment 10 Erase Counters and Registers	
<b>V0.1.15</b>		
<a href="#">Page 3</a>	Add EMEM4 and EMEM5	
<a href="#">Page 15</a>	Add BITMGR	
<a href="#">Page 15</a>	SPUCFG0 and SPUCFG1 increased from 64 Kbyte to 128 Kbyte	
<b>V0.1.16</b>		
<a href="#">Page 11</a>	TC3Ex Alternate Address Map for SOTA of Segment 8 PFLASH	
<a href="#">Page 11</a>	TC3Ex Alternate Address Map for SOTA of Segment 10 PFLASH	
<a href="#">Page 11</a>	TC3Ex Alternate Address Map for SOTA of Segment 10 Erase Counters and Registers	
<b>V0.1.17</b>		
-	No functional changes.	
<b>V0.1.18</b>		
<a href="#">Page 3</a>	CPU2.DSPR is 240 Kbyte	
<b>V0.1.19</b>		
<a href="#">Page 3</a>	CPU2.DSPR address range fixed from 240 to 96 Kbyte.	

## AURIX™ TC3xx Platform Firmware

### 3 AURIX™ TC3xx Platform Firmware

The AURIX™ TC3xx Platform Firmware from user point of view consists of the following functional parts:

- **Startup Software** (short name SSW) including **Support for Software over the Air (SOTA)**
- **Checker Software** (short name CHSW)
- Software modules implementing additional functions:
  - **Bootstrap Loaders** (short name BSL)
  - **Shutdown request handler**
  - **Power Supply Friendly Debug Monitor**

#### 3.1 Functional description

The following Sections describe the functionality of AURIX™ TC3xx Platform Firmware.

##### 3.1.1 Startup Software

The Startup Software is the first software executed after a chip reset.

SSW is executed on CPU0 - all other CPUs are kept in Halt-state during boot, to be started by user software whereas:

- SSW start address in BootROM is the reset value in Program Counter register of the CPU0. From this location an instruction is fetched and this is the first instruction executed after any device start-up.
  - immediately after this entry point the firmware checks for testmode, and in case testmode is selected - jump to test firmware is executed
- the last SSW instruction performs a jump to the first user code instruction. This first user instruction can be fetched from different locations depending on the start-up configuration as selected by the user.

The Startup Software contains procedures to initialize the device depending on one or more from the following:

- information previously stored into dedicated Flash locations
- the current state of special bits/fields in dedicated register/memory locations
- the type of event which has triggered the SSW-execution (the last reset event)
- values applied to external (configuration-) pins (optional)

The SSW can also call ASC and CAN Bootstrap Loader routines.

###### 3.1.1.1 Events triggering SSW execution

SSW execution can be triggered by different events. SSW recognizes the triggering (reset) event and takes (partially) different execution flows.

###### 3.1.1.1.1 Cold (initial) power-on reset

This is the initial powering-up of the device after the supply has been switched off, or in other words - the only way to generate this reset event is by applying power to a previously un-powered device.

The conditions at which the SSW execution starts upon this event include:

- all the registers are in their initial (reset) state
- **Flash is under reset - meaning not active, and not ready to perform any (read/erase/program) operation**
- RAMs' content is undefined
- clock system is in its initial state

## AURIX™ TC3xx Platform Firmware

Due to the overall “fully initial” state of the device upon power-on, the SSW flow is respectively longer in this case and covers the largest number of activities compared to other reset events.

### 3.1.1.1.2 System reset

From SSW point of view, the handling of system reset is generally the same as of “warm power-on”. Therefore further in this Chapter when referring to a system reset - the same SSW handling applies to warm power-on. Exceptions are noted where applicable.

This reset event can be requested by different sources:

- device internal hardware - from modules like watchdog timer and security/memory control logic
- external hardware - when active signal is applied to defined device pin(s)
- software - when defined control bit(s) are respectively installed during user code execution

For most of the sources, generating system reset is a feature configurable by software.

Applying an active low level to the PORST pin generates a system reset only if the supply voltage is above a defined level, within a defined time window. The EVR module triggers an immediate power-on reset if the supply drops below a defined level.

The conditions at which SSW execution starts upon system reset include:

- all the registers affected by these reset types (system and warm power-on) are in their initial (reset) state
- Flash is under reset - meaning not active, and not ready to perform any (read/erase/program) operation
- RAMs' content is the same as just before the system reset has been triggered
- clock system is in its initial state

### 3.1.1.1.3 Application reset

An application reset is similar to a system reset and can be requested by different sources: internal/external hardware as well as software. For all the sources, generating an application reset is a feature configurable by software.

The conditions at which SSW execution starts upon application reset include:

- registers connected to this reset type are in their initial (reset) state
- Flash is in read mode
- all the rest - RAMs and surrounding logic, the clock system and registers under system reset - is not affected by this event.

Application reset is the quickest type of reset.

### 3.1.1.2 Clock system during start-up

The state of the clock system during device start-up depends on the reset event type:

- upon power-on and system reset, the clock system is in its initial state:
  - fSRI=fCPU0 = fFSI=fBACK=100MHz nominal
  - fSPB=fSTM=fBACK/2=50MHz nominal
  - in Emulation Device (ED): fBBB=fBACK/2=50MHz nominal
  - PLL and VCO are in power-down mode
- upon application reset, the clock system does not change its state: therefore, the device runs at the same frequency and clock source as before the reset event

## AURIX™ TC3xx Platform Firmware

### 3.1.1.3 RAM overwrite during start-up

The start-up procedure can overwrite:

- up to 8kByte at the beginning of CPU0 DSPR. Therefore, this area should not be used by application software to save data which must be preserved through a reset.
- up to 1kByte at the beginning of CPU0 PSPR. Therefore this area should not be used by application software to save program code which must be preserved through a reset.

Note: *CPU0 PSPR overwrite here noted does not refer to Bootstrap Loader routines - meaning the code downloaded by BSL is properly handled and not overwritten.*

### 3.1.1.4 Stand-by controller handling during start-up

The start-up procedure will initialize stand-by controller (SCR) RAMs and trigger SCR boot when the last reset:

- was seen by the device as a cold power-on, AND
- has been triggered by EVR pre-regulator - identified by SCU\_RSTSTAT.STBYR=1

In other words, SCR is initialized/started upon cold power-on reset which is not identified as exit from stand-by mode.

Therefore, for proper SCR handling, the user software must take care to reset SCU\_RSTSTAT.STBYR flag via SCU\_RSTCON2.CLRC (refer to SCU Chapter) according to the application - e.g. after the initial system power-on, if SCR usage during stand-by mode is intended.

### 3.1.1.5 Boot Options Summary

This chapter summarizes the AURIX™ TC3xx Platform startup configurations.

For more detailed information and flow description, refer to [Chapter 3.1.1.6](#).

#### Internal Start

In this basic startup mode, the first user instruction is fetched from the Internal Program Flash of the device.

The user code start address defined as STADD is configurable in the Boot Mode Header (BMHDx.STAD) data structure.

#### Bootloader Modes

Different Bootstrap Loader routines are used in these modes to download code or data into the Program Scratchpad RAM of CPU0 (CPU0\_PSPR).

The supported Bootloader selections are:

- ASC Bootloader - ASC communication protocol via ASC pins
- Generic Bootloader via CAN pins - the communication protocol is automatically selected by the SSW between ASC and CAN

After downloading the code, the user code start address is set to the beginning of Program Scratchpad RAM (PSPR).

#### Alternate Boot Modes

In these modes (short name ABM), program code is started from an user-defined address, but only if all defined conditions are satisfied (see [Chapter 3.1.1.6.2](#)). If the conditions are not met, the SSW can fall back to a Bootstrap Loader mode, to allow download and execution of user code.

## AURIX™ TC3xx Platform Firmware

All the information needed for the SSW to handle ABM startup mode is collected into Alternate Boot Mode Headers (ABMHD). The checks are performed according to [Alternate Boot Mode evaluation](#) flow as defined in [Chapter 3.1.1.6.2](#).

If this mode is selected and the ABMHD is valid, the user code start address STADD is set to the respective value from the Alternate Boot Mode Header (ABMHDn.STADABM).

### 3.1.1.6 Boot Mode evaluation sequence

This is a main functional part of AURIX™ TC3xx Platform [Startup Software Main Flow](#) - see also [Figure 22](#) and [Chapter 3.1.1.7.4](#)

Boot Mode evaluation follows the sequence shown on [Figure 18](#) and [Figure 19](#), where three blocks can be differentiated as shown in the next Sections.

**Note:** *All the CRC calculations below pointed are done using the TriCore instruction CRC32 which implements the IEEE 802.3 standard: CRC-32 polynomial is used and the CRC result is bit-reversed and inverted.*

#### 3.1.1.6.1 Evaluation of Boot Mode Headers

A Boot Mode Header is an information structure (see [Table 46](#)) that specifies user startup options, including the starting address of user code.

In AURIX™ TC3xx Platform, [four sets of Boot Mode Headers](#) (BMHDx for x=0, 1, 2, 3) are defined in the User Configuration Blocks (UCB) of Data Flash (DFLASH). Each set contains [an original and copy](#) in UCB\_BMHDx\_ORIG and UCB\_BMHDx\_COPY, respectively.

**Note:** *For full content and layout description - refer to the “User Configuration Block (UCB)” Section in the “Non Volatile Memory (NVM) Subsystem” Chapter.*

## AURIX™ TC3xx Platform Firmware

**Table 46 Boot Mode Header (BMHD) structure**

Field Name	Subfield	Description
BMI	Boot Mode Index - 16 bit	
	PINDIS bit [0]	Mode selection by configuration pins: <b>0B Mode selection by HWCFG pins is enabled</b> 1BMode selection by HWCFG pins is disabled
	HWCFG bits [3:1]	Start-up mode selection: <b>111B Internal start from Flash</b> 110BAAlternate Boot Mode (ABM) 100B Generic Bootstrap Loader Mode (ASC/CAN BSL) 011BASC Bootstrap Loader Mode (ASC BSL) elseinvalid
	LSENA0 bit [4]	Lockstep monitoring control by SSW for CPU0: 0B Lockstep monitoring for CPU0 is disabled <b>1BLockstep monitoring for CPU0 is enabled</b>
	LSENA1 bit [5]	Lockstep monitoring control by SSW for CPU1: <sup>1)</sup> 0B Lockstep monitoring for CPU1 is disabled <b>1BLockstep monitoring for CPU1 is enabled</b>
	LSENA2 bit [6]	Lockstep monitoring control by SSW for CPU2: <sup>1)</sup> 0B Lockstep monitoring for CPU2 is disabled 1BLockstep monitoring for CPU2 is enabled
	LSENA3 bit [7]	Lockstep monitoring control by SSW for CPU3: <sup>1)</sup> 0B Lockstep monitoring for CPU3 is disabled 1BLockstep monitoring for CPU3 is enabled
	LBISTENA bit [8]	LBIST execution start by SSW: 0B LBIST execution start by SSW is disabled <b>1B LBIST execution start by SSW upon cold power-on is enabled</b>
	CHSWENA bits [11:9]	Checker Software (CHSW) execution after SSW: <sup>2)</sup> <b>101B CHSW execution after SSW is disabled</b> else CHSW execution after SSW is enabled
	reserved bits [15:12]	Reserved for future extensions, must be configured to 0 in UCB_BMHDx
BMHDID	---	Boot Mode Header Identifier - 16 bit: <b>B359H</b> BMHDID OK elseBMHDID invalid
STAD	---	Start address (always must be inside PFLASH, word-aligned) - 32 bit: if ABM selectedStart address of the Alternate Boot Mode Header if Internal start selectedStart address of the user code elsenot considered for mode selection
CRCBMHD	---	Check result for the Boot Mode Header - 32 bit
CRCBMHD_N	---	Inverted check result for the Boot Mode Header - 32 bit

- 1) Only if the respective CPUx Lockstep functionality is available on the product, otherwise the bit is Reserved, must be configured to 0 in UCB\_BMHDx
- 2) This bitfield is not evaluated during the SSW flow but by the **Checker Software** - refer to **Chapter 3.1.2**

## AURIX™ TC3xx Platform Firmware

**Attention:** *The requirement in the above Table - STAD to be a valid word-aligned address inside PFLASH (considering the complete PFLASH, not PF0 only) - is always applicable, meaning:*

- *independently which mode is selected - also in case of BSL mode when the value has no effect*
- *independently how is the mode selected - also in case of ABM or Internal start mode selected by pins*

**Attention:** *For the above requirement, both PFLASH address areas for cached and non-cached access are considered as allowed locations by the evaluation procedure.*

The SSW follows these steps to process the Boot Mode Headers and their copies (refer to [Figure 18](#)):

1. set SSW variables/flags to these initial states:
  - a) no valid BMI found usable for boot
  - b) no valid start-up configuration found usable for boot (BOOT\_CFG)
  - c) no configuration from pins selected for boot (BOOT\_PIN)
2. check the status of the currently evaluated UCB\_BMHDX - evaluate bitfields in DMU\_HF\_CONFIRM0 register as follows:
  - for BMHD[n] originals - into PROINBMHDnO bitfields
  - for BMHD[n] copies - into PROINBMHDnC bitfields
  - a) **if the target bitfield indicates status CONFIRMED (10B) or UNLOCKED (01B) - continue with the next step**
  - b) otherwise - the status of current BMHDx is wrong, go to step 17.
3. check the Boot Mode Header Identifier (BMHDID) value against the value in [Table 46](#)
  - a) if value OK - continue with the next step
  - b) otherwise - BMHDID of the current BMHDx is wrong, go to step 17.
4. check the Boot Mode Index (BMI) value against [Table 46](#) - HWCFG subfield must be valid, all undefined bits zero
  - a) if value OK - continue with the next step
  - b) otherwise - BMI of the current BMHDx is wrong, go to step 17.
5. check the Start Address (STAD) value - it must be a valid word-aligned address inside PFlash
  - a) if value OK - continue with the next step
  - b) otherwise - STAD in the current BMHDx is wrong, go to step 17.
6. calculate CRC over the BMHDx content (total 2 words including BMI, BMHDID and STAD) and compare against CRCBMHD value stored inside BMHDx, then invert the calculated value and compare the result against CRCBMHD\_N value from BMHDx
  - a) if both values OK - continue with the next step
  - b) otherwise - the current BMHDx is wrong, go to step 17.
7. check if Boot Mode selection from pins is enabled by the current BMHDx (BMI.PINDIS=0) and in general for the device (DMU\_HF\_PROCONTP.BML=00B)
  - a) if yes - continue with the next step
  - b) otherwise - only configuration from BMI is allowed, go to step 10.
8. check if Boot Mode selection from pins is enabled by low level (zero) latched at HWCFG3 pin upon reset (SCU\_STSTAT.HWCFG[3]=0)
  - a) if yes - continue with the next step
  - b) otherwise - configuration from BMI will be done, go to step 10.
9. prepare Boot Mode selection from pins (the selection still can be canceled) - refer to [Table 47](#)

## AURIX™ TC3xx Platform Firmware

- a) instal the index of current BMHD into BMHD\_INDEX
  - b) instal the mode selection according to SCU\_STSTAT.HWCFG[5:4] into BOOT\_CFG (see the Note below)
  - c) set “configuration from pins” flag into BOOT\_PIN
  - d) go to step 11.
10. prepare Boot Mode selection from BMI (the selection still can be canceled)
- a) instal the index of current BMHD into BMHD\_INDEX
  - b) instal the mode selection according to BMHDx.BMI into BOOT\_CFG (see the Note below)
11. check if Alternate Boot Mode is so far selected for start-up according to BOOT\_CFG
- a) if yes - continue with the next step
  - b) otherwise - no ABM (meaning no further evaluation needed), go to step 13.
12. evaluate Alternate Boot Mode Headers and the user code according to the procedure described in  
**Chapter 3.1.1.6.2**
- a) if procedure is successfully completed - go to step 14.
  - b) otherwise - ABM evaluation failed, go to step 17.
13. check if a Bootstrap Mode is selected according to BOOT\_CFG
- a) if yes - instal the first address of CPU0 Program scratchpad RAM (CPU0\_PSPR) as user code start address into BOOT\_ADDR
  - b) otherwise - instal BMHDx.STAD value as user code start address into BOOT\_ADDR
14. save boot mode information - to be available for HSM and application software
- a) selected boot mode BOOT\_CFG, the index of the valid BMHD BMHD\_INDEX, the flag ORIG/COPY BMHD\_COPY and pin-configuration flag BOOT\_PIN - into **SCU\_STMEM1**
  - b) user code start address BOOT\_ADDR - into SCU\_STMEM2[31:2] (the address is word aligned - bits[1:0] are not changed here), it is to be used also in case of CPU0 kernel reset
15. configure lockstep monitoring feature from the valid BMI found - instal BMI.LSENAn bits into respective LSENn bits of SCU\_LCLCON0/1 registers (n=0,1,2,3)
16. set “BMI\_VALID” and BOOTMODE\_CONFIGURED flags in **SCU\_STMEM1** register and exit the sequence
- 17.
18. according to the currently evaluated BMHDx
- a) if ORIGINAL - switch evaluation to the COPY and restart the sequence from step 1.
  - b) otherwise - exit the sequence (no valid BMHD found so far)

**Attention:** *SSW flow as above described allows evaluation both of UCB\_BMHDx ORIGINAL and COPY, but in reality always only one of these two UCBs can be indicated at any point of time in DMU\_HF\_CONFIRM0 register as being CONFIRMED or UNLOCKED - for more information, refer to DMU Section in “Non Volatile Memory (NVM) Subsystem” Chapter.*

**Therefore:**

- if the ORIGINAL of a given UCB\_BMHDx has been evaluated, the flow will continue with the COPY but it will be immediately exited due to state being different from CONFIRMED or UNLOCKED
- the COPY will be only evaluated if the ORIGINAL is in a state different from CONFIRMED or UNLOCKED.

**Note:** *BOOT\_CFG is installed not (always) directly from BMI or HWCFG but according to the coding in **SCU\_STMEM1** register description.*

## AURIX™ TC3xx Platform Firmware

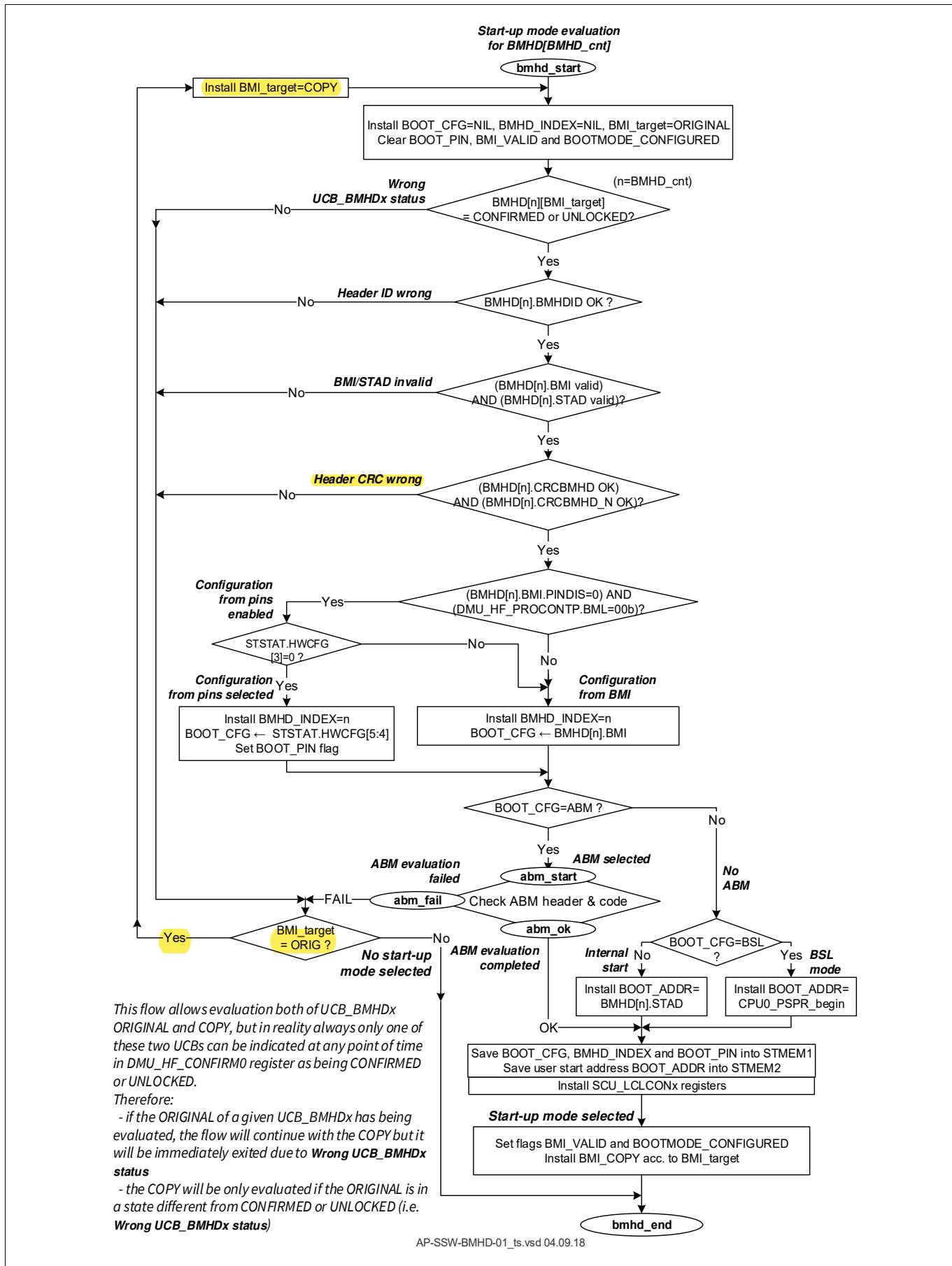


Figure 18 Boot Mode Headers evaluation flow

## AURIX™ TC3xx Platform Firmware

**Table 47 Start-up mode selection by pins in AURIX™ TC3xx Platform**

HWCFG pins		Start-up Mode
[5]	[4]	
1	1	Internal start from Flash
1	0	ABM, Generic Bootstrap Loader on fail
0	1	ABM, ASC Bootstrap Loader on fail
0	0	Generic Bootstrap Loader

### 3.1.1.6.2 Alternate Boot Mode evaluation

Alternate Boot Mode handling is controlled by the content of Alternate Boot Mode Headers. The structure of a single Alternate Boot Mode Header is defined in [Table 48](#).

**Table 48 Alternate Boot Mode Header (ABMHD) structure**

Offset Address	Size Byte	Field Name	Description
00H	4	STADABM	User Code Start Address in ABM mode <sup>1)</sup>
04H	4	ABMH DID	Alternate Boot Mode Header Identifier: FA7C B359H ABMH DID OK else ABMH DID invalid
08H	4	CHKSTART	Memory Range to be checked - Start Address <sup>1)</sup>
0CH	4	CHKEND	Memory Range to be checked - End Address <sup>1)</sup>
10H	4	CRCRANGE	Check Result for the Memory Range
14H	4	CRCRANGE_N	Inverted Check Result for the Memory Range
18H	4	CRCABMHD	Check Result for the ABM Header
1CH	4	CRCABMHD_N	Inverted Check Result for the ABM Header

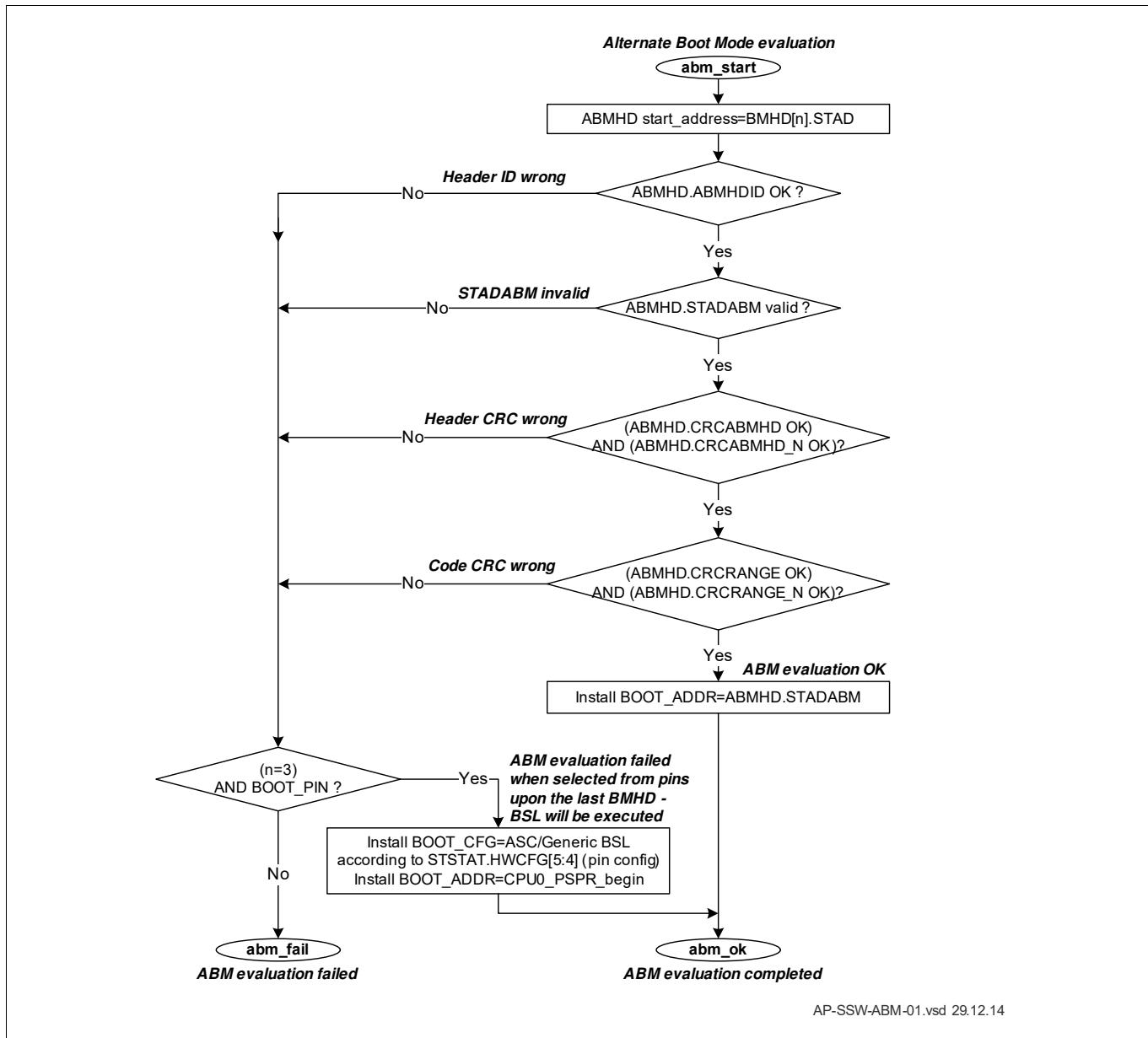
1) All the addresses must be inside PFLASH (considering the complete PFLASH, not PF0 only), word-aligned

In AURIX™ TC3xx Platform Alternate Boot Mode Header(s) can be located starting at arbitrary word-aligned locations inside PFlash. The start address is taken from the STAD field of the Boot Mode Header selecting ABM (refer to [Table 46](#))

**Attention:** Both the PFLASH address areas for cached and non-cached access are considered as allowed locations by this evaluation procedure, whereas both CHKSTART and CHKEND must be either in cached or non-cached segment - mixing not allowed.

Alternate Boot Mode Header (ABMHD) evaluation flow (refer to [Figure 19](#)) includes the following steps:

## AURIX™ TC3xx Platform Firmware

**Figure 19 Alternate Boot Mode evaluation flow**

1. check the Alternate Boot Mode Header Identifier (ABMH DID) value against the value in [Table 48](#)
  - a) if value OK - continue with the next step
  - b) otherwise - ABMH DID is wrong, go to step 6.
2. check the Start Address (STADABM) value - it must be a valid word-aligned address inside PFlash
  - a) if value OK - continue with the next step
  - b) otherwise - STADABM is wrong, go to step 6.
3. calculate CRC over the ABMHD content (total 6 words including STADABM, ABMH DID, CHKSTART, CHKEND, CRCRANGE and CRCRANGE\_N) and compare against CRCABMHD value stored inside ABMHD, then invert the calculated value and compare the result against CRCABMHD\_N value from ABMHD
  - a) if both values OK - continue with the next step
  - b) otherwise - ABMHD is wrong, go to step 6.

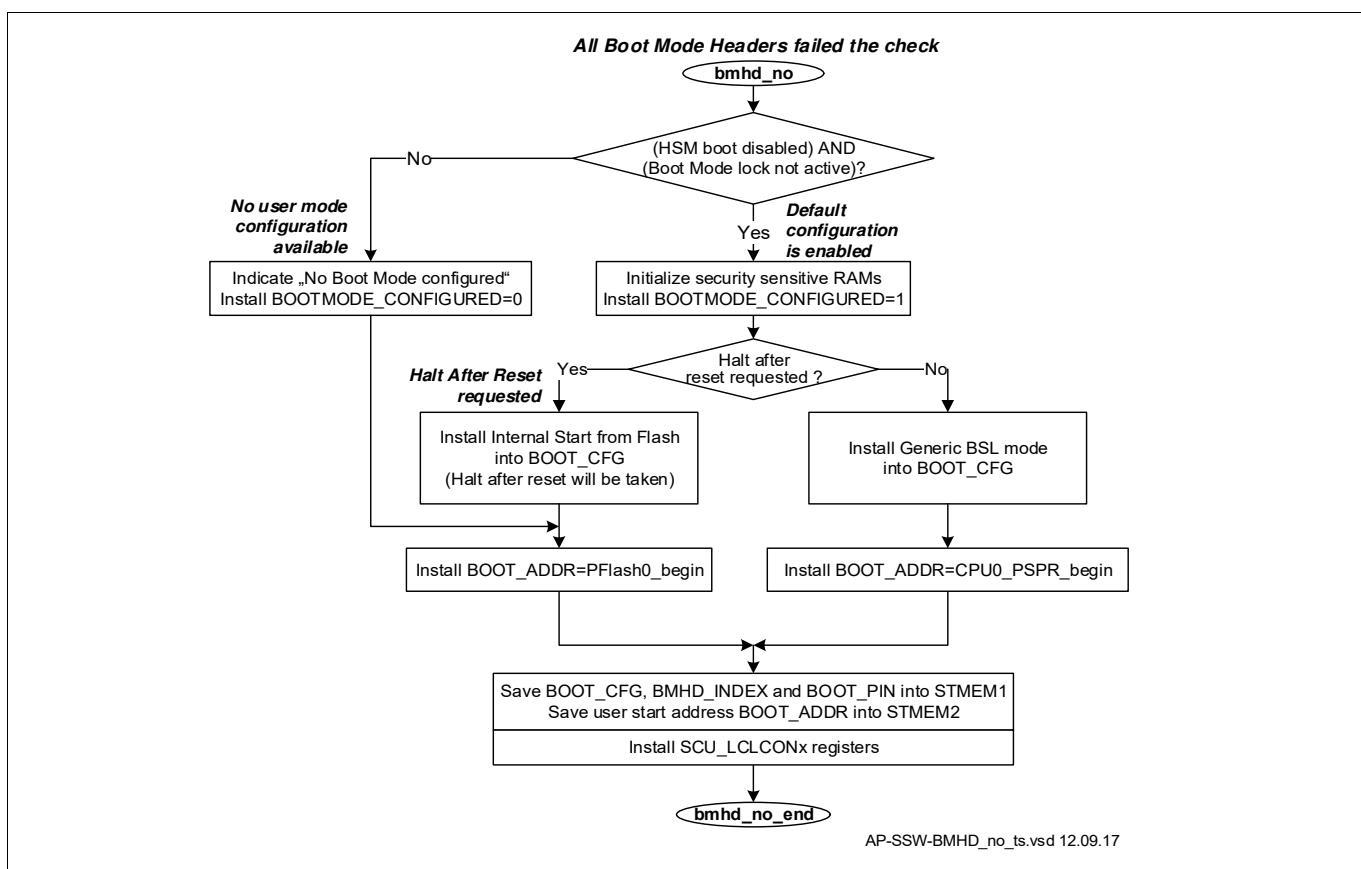
## AURIX™ TC3xx Platform Firmware

4. calculate CRC over the memory area starting with CHKSTART and ending with CHKEEND (word location) and compare against CRCRANGE value stored inside ABMHD, then invert the calculated value and compare the result against CRCRANGE\_N value from ABMHD
  - a) if both values OK - continue with the next step
  - b) otherwise - user code is wrong, go to step 6.
5. instal ABMHD.STADABM value as user code start address into BOOT\_ADDR, then exit ABM evaluation sequence as successfully completed - ABM will be effectively executed
6. check if ABM was selected from pins (BOOT\_PIN set by **Evaluation of Boot Mode Headers** sequence) during evaluation of the last BMHDn (n=3)
  - a) if yes - continue with the next step
  - b) otherwise - exit ABM evaluation sequence as failed
7. configure BSL start-up mode for execution
  - a) instal BOOT\_CFG to ASC or Generic BSL mode according to STSTAT.HWCFG[5:4] (refer to **Table 47**)
  - b) instal the first address of CPU0 Program scratchpad RAM (CPU0\_PSPR) as user code start address into BOOT\_ADDR
  - c) exit ABM evaluation sequence as successfully completed - the selected BSL will be effectively executed

### 3.1.1.6.3 Processing in case no valid BMHD found

If no valid Boot Mode Header was found by the above sequences, the SSW does not execute user code. The SSW prepares the device so that the user can connect, install valid Boot Mode Headers, program application code into PFlash, or install other device configurations.

In such a case, the SSW flow follows **Figure 20**:



**Figure 20 Flow when no valid BMHD found**

## AURIX™ TC3xx Platform Firmware

1. check if “default configuration” is enabled for the device - the condition is HSM boot disabled (DMU\_SP\_PROCONHSMCCFG.HSMBOOTEN=0) and Boot Mode Lock not active (DMU\_HF\_PROCONT.P.BML=00B)
  - a) if yes - set flag BOOTMODE\_CONFIGURED=1 in **SCU\_STMEM1** register, continue with the next step
  - b) if not - indicate “No Boot Mode configured” (reset flag BOOTMODE\_CONFIGURED in **SCU\_STMEM1** register) and go to step 4.

NOTE: In such a case, **Processing in case no Boot Mode configured by SSW** will be executed.
2. initialize security sensitive RAMs
3. For definition which RAMs are security-sensitive - refer to “Memory Test Unit” Chapter.
4. check if Halt After Reset request has been received (CBS\_OSTATE.HARR=1)
  - a) if yes - instal BOOT\_CFG to Internal start from Flash mode
  - b) this mode will not be effectively taken by SSW immediately at its end; halt after reset will be executed instead.
  - c) if not - instal BOOT\_CFG to Generic BSL mode, instal the first address of CPU0 Program scratchpad RAM (CPU0\_PSPR) as user code start address into BOOT\_ADDR, then go to step 5.
5. instal the address with offset 0x0020 in logical sector S40 in PFLASH0 (I.e. 0xA000A020) as user code start address into BOOT\_ADDR
  - a) this start address will not be effectively taken by SSW - either halt after reset will be executed, or SSW will enter an endless loop (see **Chapter 3.1.1.7.9**)
6. save boot mode information so it is available for HSM and application software
  - a) selected boot mode BOOT\_CFG - into SCU\_STMEM1
  - b) the installed start address BOOT\_ADDR - into SCU\_STMEM2[31:2] (the address is word aligned - bits[1:0] are not changed here)
7. configure lockstep monitoring feature as when no valid BMI is found - instal SCU\_LCLCON0.LSEN0=0 (disable lockstep for CPU0)

### 3.1.1.6.4 Processing in case no Boot Mode configured by SSW

If no valid Boot Mode Header was initially found and also the above sequence (**Processing in case no valid BMHD found**) was exited on its first step (because “default configuration” is disabled for the device) - the SSW can not select any start-up mode, meaning no exit from SSW will be effectively taken.

In such a case SSW flow is (refer to **Figure 21**):

- check if debug access is allowed for the device
  - if yes - enable debug access by installing CBS\_OSTATE.IF\_LCK to 1 (done using CBS\_OEC register)
- enter endless NOP loop

## AURIX™ TC3xx Platform Firmware

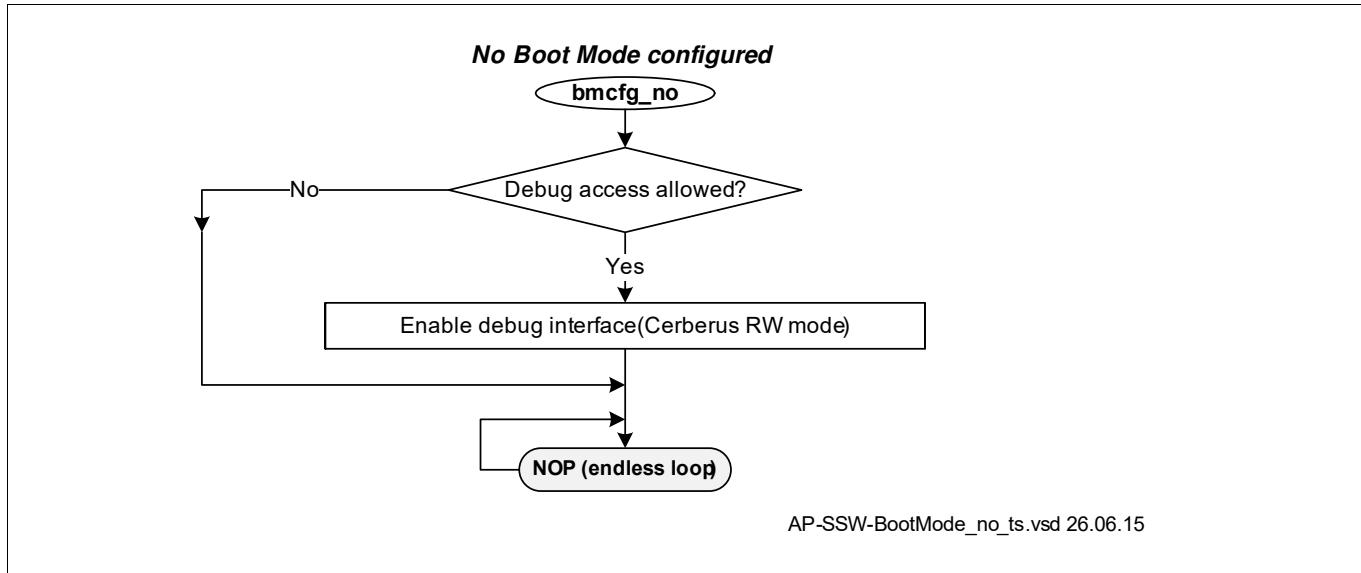


Figure 21 Flow when no boot mode configured

### 3.1.1.7 Startup Software Main Flow

Figure 22 shows the execution steps of the SSW:

## AURIX™ TC3xx Platform Firmware

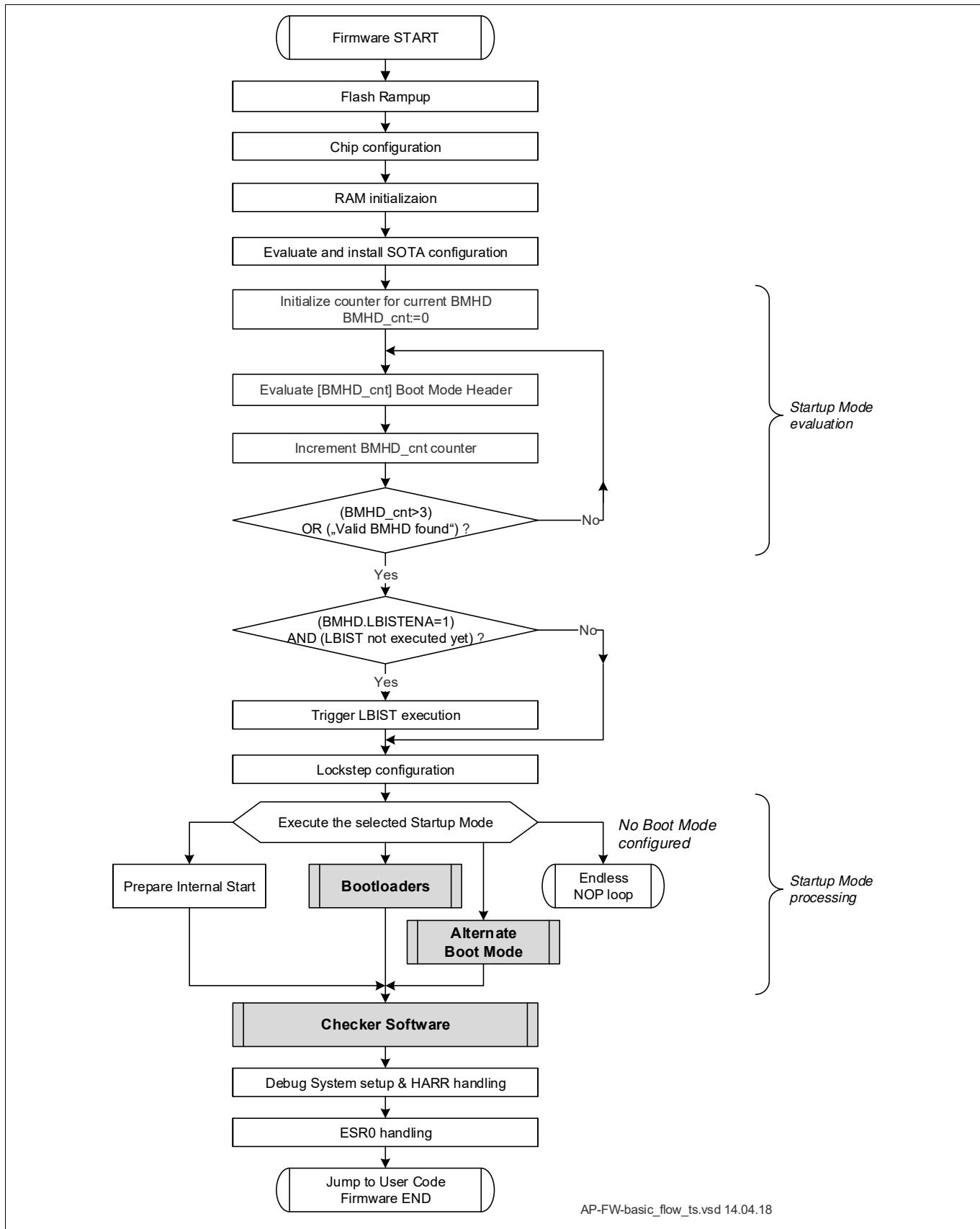


Figure 22 AURIX™ TC3xx Platform Firmware: main flow

## AURIX™ TC3xx Platform Firmware

### 3.1.1.7.1 Flash ramp-up

In this SSW module, the Flash is enabled and brought to a state where all the operations (read, erase, program and command sequences) can be performed within the full range of specified working conditions.

### 3.1.1.7.2 Device Configuration

The target of this SSW module is to initialize a number of AURIX™ TC3xx Platform registers.

### 3.1.1.7.3 RAM Initialization

The AURIX™ TC3xx Platform SSW supports RAM initialization, which is user-configurable by programming sections within UCB\_DFLASH, from which the configuration is installed during start-up into the DMU\_HF\_PROCONRAM register.

*Note:* For content and layout description, refer to the “User Configuration Block (UCB)” and the “Data Memory Unit (DMU)” Sections in the “Non Volatile Memory (NVM) Subsystem” Chapter.

The initialization procedure fills selected RAMs with all zero data content and also installs correct ECC bits, so that accesses to initialized RAMs shall not produce an uncorrectable ECC error.

The following configuration options are available:

- optional initialization upon cold and/or warm power-on of the device
- optional initialization for the RAMs of any separate CPU
- optional initialization for the RAM of any separate Local Memory Unit (LMU)

The RAMs supporting stand-by mode - CPU0\_DLMU and CPU1\_DLMU - are not initialized after wake-up even when selected in DMU\_HF\_PROCONRAM register, if RAM(s) being supplied during stand-by - the conditions checked are:

- for CPU0\_DLMU - PMS\_PMSWSTAT2.STBYRAM=x1xB or xx1B
- for CPU1\_DLMU - PMS\_PMSWSTAT2.STBYRAM=1xxB

### 3.1.1.7.4 Select and execute Startup Modes

The AURIX™ TC3xx Platform SSW evaluates up to four Boot Mode Headers (BMHD) sequentially, each of which may have an Original and a Copy in Data Flash UCBs.

For AURIX™ TC3xx Platform User Startup Configurations and modes refer to [Chapter 3.1.1.5](#). For the flow of evaluating any single BMHD - to [Chapter 3.1.1.6](#).

Once a BMHD evaluation is successfully completed, the selected mode is taken, otherwise: [Processing in case no valid BMHD found](#) or [Processing in case no Boot Mode configured by SSW](#) is executed.

#### Secure Boot option handling

If HSM module is available (according to device configuration), Secure Booting is supported, in which the user code is processed by the HSM.

The SSW supplies information about the current device mode, reset type, the primary selected start-up mode and the effectively taken start-up mode. This information is then available then for the HSM module for boot and user code processing.

### 3.1.1.7.5 LBIST execution

If selected in a valid BMHD by LBISTENA=1 (refer to [Chapter 3.1.1.6](#)), LBIST execution will be triggered by SSW.

## AURIX™ TC3xx Platform Firmware

The configuration applied corresponds to the “LBIST Configuration A” as defined in “LBIST considerations for TC3xx” Section of the product specific Appendix to the Target Specification.

If triggered, LBIST will end with an internal reset, leading to a new execution of the SSW. During this execution, in case of successful LBIST completion (SCU\_LBISTCTRL0.LBISTDONE=1) SSW will not re-trigger LBIST but exit to the user code.

Note, that SSW does not evaluate the result from LBIST execution (in SCU\_LBISTCTRL3 register) - this is to be done by the application software.

### 3.1.1.7.6 Lockstep configuration

Upon cold power-on only, the SSW performs Lockstep configuration as follows:

- if a valid BMI has been found during start-up mode evaluation:
  - lockstep control (for CPUs supporting lockstep) is installed from BMI.LSENAn bits into respective LSENN bits of SCU\_LCLCON0/1 registers (n=0,1,2,3)
- otherwise
  - lockstep for CPU0 is disabled by installing SCU\_LCLCON0.LSEN0=0

### 3.1.1.7.7 Debug System handling

The SSW internal flag Unlock Debug Interface is set to control debug access to the device, according to the following evaluation sequence:

1. The SSW checks whether an external tool has requested debug access by writing a defined content (32-bit value CMD\_KEY\_EXCHANGE) into COMDATA register
  - a) if yes - continue with the next step
  - b) if not - go to step 4.
2. While still in Cerberus Communication mode, the SSW confirms request reception and receives 8 further words from the COMDATA register
3. the data received (256 bits) is sent by SSW to DMU to be checked as debug interface password, and the result is evaluated by SSW:
  - a) if OK - debug password is correct, set SSW internal flag, debug interface will be unlocked, exit the sequence
  - b) otherwise - continue with the next step
4. instal into COMDATA a 32-bit value serving for an external tool to identify the device connected - UNIQUE\_CHIP\_ID\_32BIT

*Note: The name here used (UNIQUE\_CHIP\_ID\_32BIT) should be not misleading - the value considered and written into COMDATA register is NOT identifying uniquely any single device, but the product variant only. In case chip-unique identification is needed, the user Software (or the tool) should read the Unique Chip Identifier from UCB\_USER - refer to the “User Configuration Block (UCB)” Section of the “Non Volatile Memory (NVM) Subsystem” Chapter.*

5. check if Flash read protection is activated:
  - a) if yes - debug interface will be left locked, no debug access to device, exit the sequence
  - b) if not - set SSW internal flag, debug interface will be unlocked, exit the sequence

*Note: If the Debug Interface is configured as locked by the DMU\_HF\_PROCONDBG.DBGIFLCK=1 (bit installed during start-up according to the DFlash UCB\_DBG content), debug access for a device with installed Flash read protection shall be only allowed if correct debug password is received (as of steps 1. to 3. above).*

## AURIX™ TC3xx Platform Firmware

Next, Halt after Reset is prepared if requested. The SSW processing here is as follows:

- check whether external (debug) access to the device will be generally granted
  - if Not -> exit this procedure
- check whether Halt After Reset is requested
  - if Not -> exit this procedure
- configure a Break Before Make breakpoint at the first user code instruction
- enable On-Chip Debug Support system.

### 3.1.1.7.8 ESR0 pin handling

If both of these conditions

- ESR0CNT<=>FFFFH in DMU\_HF\_PROCONDF register (refer to register description in “Data Memory Unit” Section of “Program Memory Unit” Chapter) AND
  - ESR0-pin configuration upon SSW entry is open-drain reset output (SCU\_IOCR.PC0 in [1110B, 1101B])
- are satisfied, SSW will:

- release ESR0 pin - by installing SCU\_ESROCFG.ARC:=1, which clears the Application Reset Indicator in SCU\_ESROCFG.ARI with a configurable delay after device internal reset is released (i.e. CPU0 started). The delay is defined as follows
  - if DMU\_HF\_PROCONDF.ESR0CNT=000H - about 500μsec upon cold power-on, not longer than 40μsec otherwise
  - if DMU\_HF\_PROCONDF.ESR0CNT=001H...FFEH - (ESR0CNT)\*10μsec, where:
    - \* upon cold power-on the minimum possible delay is about 500μsec
    - \* otherwise - the minimum possible delay is about 40μsec
- wait until ESR0 pin is effectively high (indicated by SCU\_IN.P0=1) at the very SSW end, before jumping to the first user instruction

To generate the configurable ESR0 delay after device reset release, the SSW uses System Timer 0 (STM0), and takes into account the following default settings after power-on/system reset:

- STM is reset and starts counting from zero
- STM is clocked with 50MHz i.e. fSTM=fBACK/2

**Attention:** Both the above conditions could be not true after application reset, if default settings are changed by user code executed after power-on/system reset. In such a case, ESR0 handling by SSW will not work correctly, meaning the real prolongation will not correspond to ESR0CNT value configured.

### 3.1.1.7.9 Ending the SSW and Starting the User Code

The last steps executed by the SSW are:

- unlock (in case) Debug Interface
- jump to the first User Instruction at address STADD.

Additionally, if all the following conditions are satisfied:

- the device is an ED
- debug access to device is allowed
- halt after reset is not requested
- the last reset has been a power-on (cold or warm)

## AURIX™ TC3xx Platform Firmware

SSW performs at its very end additional checks and in case all these succeed - the last SSW instruction jumps not to the “standard” STADD but to an address inside EMEM (Emulation Memory). This SSW part implements the sequence defined in AURIX™ TC3xx Platform ED Target Specification, Chapter “Startup with prolog code in EMEM”.

### 3.1.2 Checker Software

The Checker Software (CHSW) is intended to evaluate whether the device configuration and preparation for user code execution - in particular the aspects considered as safety relevant - are correct after completion of the Start-up Software.

#### 3.1.2.1 CHSW execution flow

CHSW (besides the below checks as initial parts from it) is only executed if all the following conditions are fulfilled:

1. Bootmode evaluation done by SSW is found to be correct

To decide on this, CHSW evaluates and compares the boot information stored by SSW into the **Registers providing information on the boot selections** (refer to [Chapter 3.2.1.1](#)) against the boot information from the respective Boot Mode Header UCB (UCB\_BMHDX) area in DFLASH, also in case - hardware configuration pins (HWCFG[5:3]) and other start-up related factors as described in [Chapter 3.1.1.6](#)

2. Valid Boot Mode Index (BMI) has been found and used for start-up mode selection

To remind, in many cases the device may continue code execution after start-up even if no BMHD/BMI has been found by SSW - refer to [Chapter 3.1.1.6.3 Processing in case no valid BMHD found](#). In those scenarios, the (rest of) CHSW is not executed.

3. **The Boot Mode Index taken for start-up does not disable CHSW execution**

The above condition is fulfilled if the BMI taken for start-up contains CHSWENA bitfield not equal to 101B (refer to [Table 46](#)).

## AURIX™ TC3xx Platform Firmware

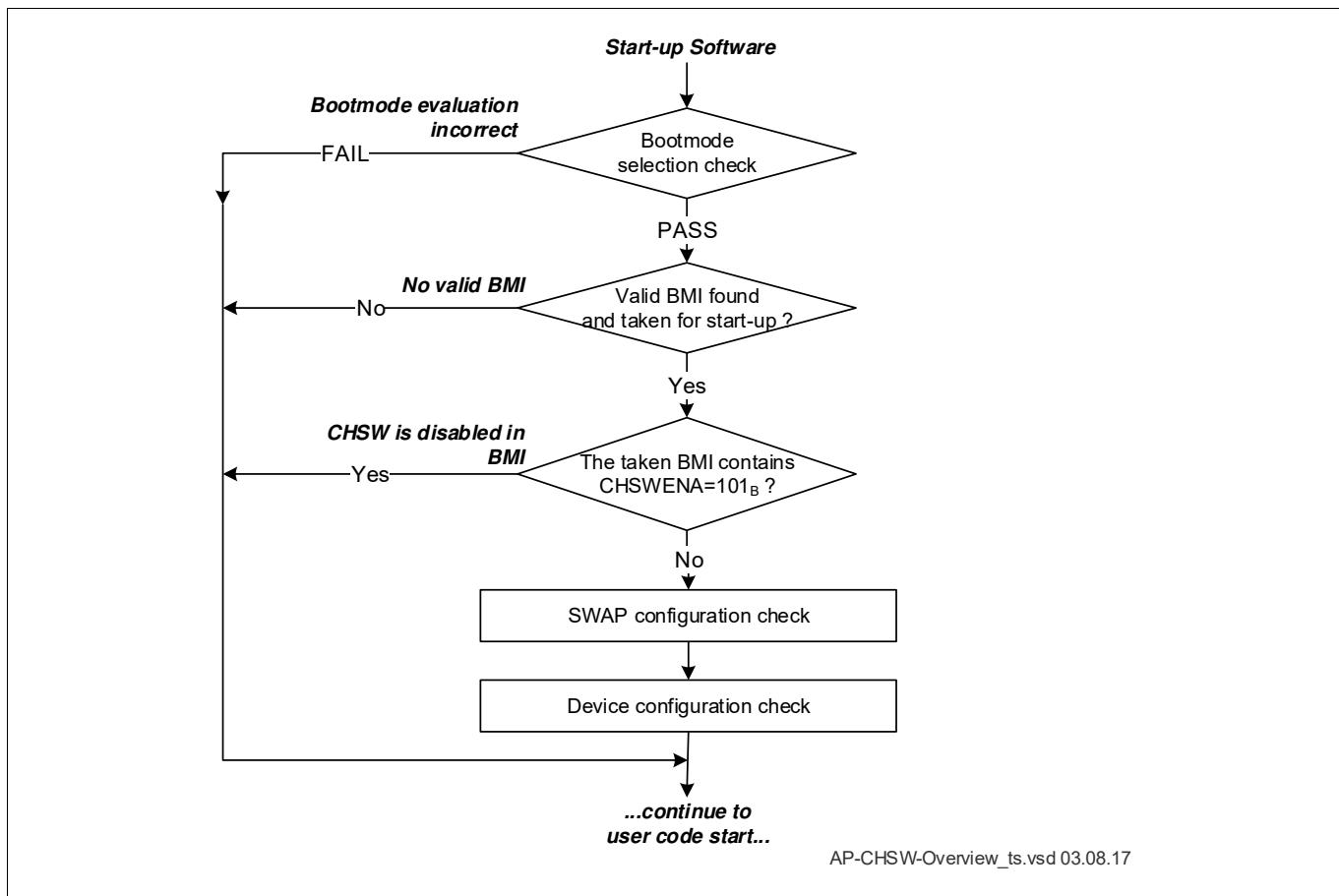


Figure 23 AURIX™ TC3xx Platform Checker Software Overview

If all the above conditions are met - CHSW executes the main device checks as described in the following Section.

### 3.1.2.2 Checks performed by CHSW and exit information

The checks supported by CHSW in AURIX™ TC3xx Platform are listed in [Table 49...Table 53](#) below, to note:

- most of the checks are executed upon defined reset types only
- (some of) the checks are only executed on products implementing the target module to be checked - e.g. devices with EMEM, RIF module, GETH module(s) etc. In particular, EMEM-related checks are only performed on devices implementing Emulation Memory and only for those EMEM module(s) which are effectively present on the product.

CHSW communicates the checks' execution status and result to the user software via the [Registers providing information on the Checker Software activity](#) (refer to [Chapter 3.2.1.2](#)), whereas:

- bits [0] in all the registers SCU\_STMEM3...SCU\_STMEM6 are set to 1 upon user code start
- anyone of the other used bits in all the registers is assigned to a given check supported by CHSW as shown in [Table 49...Table 53](#)
- the bits in any register indicate defined status of the check as follows:
  - SCU\_STMEM3** - the check is started
  - SCU\_STMEM4** - the check failed
  - SCU\_STMEM5** - the check passed

## AURIX™ TC3xx Platform Firmware

- **SCU\_STMEM6** - the check is finished

Following from the above, the overall check status should be interpreted by user software according to **Table 54**.

**Note:** *Device start-up after LBIST execution is handled by TC3xx Firmware as cold power-on, respectively the CHSW results indicated in such a case correspond to the checks executed upon this type of device reset.*

**Table 49 Checks of the reset evaluation**

<b>Check of:</b>	<b>Assigned bits in SCU_STMEM3...6</b>	
	<b>position</b>	<b>name (excluding _C*)</b>
Reset type evaluation upon cold power-on (acc. to CHSW)	4	CPOR
Reset type evaluation upon warm power-on (acc. to CHSW)	5	WPOR
Reset type evaluation upon system reset (acc. to CHSW)	6	SYSR
Reset type evaluation upon application reset (acc. to CHSW)	7	APPR

**Table 50 Checks executed by CHSW upon any device reset**

<b>Check of:</b>	<b>Assigned bits in SCU_STMEM3...6</b>	
	<b>position</b>	<b>name (excluding _C*)</b>
General CHSW execution status	1	CHSWGGEN
Bootmode selection	2	BMSEL
SWAP configuration	3	SWAP
Shutdown request handler entry point (SCU_RSTCON3 register)	11	RSTCON3
Gigabit Ethernet MAC module calibration	21	GETH <sup>1)</sup>
Gigabit Ethernet MAC second module calibration	24	GETH1 <sup>1)</sup>
RIF0 module calibration	22	RIF0
RIF1 module calibration	23	RIF1
PFLASH uncorrectable ECC control settings (CPUx_FLASHCON1 register)	29	FLASHCON1

- 1) The check for Gigabit Ethernet MAC module(s) calibration will fail after application reset, if the application software has not enabled GETH clock (in SCU\_CCUCON5 register) after the previous power-on/system reset(s) - meaning if the module is not used by application - therefore in such use-case anyway the check for this' module calibration is not relevant.

**Table 51 Checks executed by CHSW upon cold power-on only**

<b>Check of:</b>	<b>Assigned bits in SCU_STMEM3...6</b>	
	<b>position</b>	<b>name (excluding _C*)</b>
EVR and power-system trimming - check #1	8	EVRT1
EVR and power-system trimming - check #2	9	EVRT2
Converter control block trimming	20	CONVCTRL

## AURIX™ TC3xx Platform Firmware

**Table 52 Checks executed by CHSW upon cold and warm power-on**

Check of:	Assigned bits in SCU_STMEM3...6	
	position	name (excluding _C*)
System & Peripheral PLLs trimming	14	PLLTRIM
FM amplitude trimming	31	FMTRIM
ECC control for EMEM0 and EMEM3 - check #1 (MC44_ECCS, MC47_ECCS registers)	16	EMEMT1
ECC control for EMEM0 and EMEM3 - check #2 (MEMCON registers for EMEM0 and EMEM3 - s0, s3)	17	EMEMT2

**Table 53 Checks executed by CHSW upon system reset, cold and warm power-on**

Check of:	Assigned bits in SCU_STMEM3...6	
	position	name (excluding _C*)
Chip identification (SCU_CHIPID register)	12	CHIPID
Clock system main settings (CCU_CCUCON0 register)	13	CCUCON0
PFLASH wait states settings (DMU_HF_PWAIT register)	15	PWAIT

**Table 54 Status indication by CHSW**

Register.bits	Check status		
	not executed	PASS	FAIL
SCU_STMEM3.*_CS	0	1	1
SCU_STMEM4.*_CF	0	0	1
SCU_STMEM5.*_CP	0	1	0
SCU_STMEM6.*_CE	0	1	1

### 3.1.2.3 Checker Software exit information for ALL CHECKS PASSED

SCU\_STMEM3...SCU\_STMEM6 registers' content corresponding to "ALL CHECKS PASSED" result from Checker Software upon different reset types is provided by Firmware Chapter of the device specific "Appendix" document.

### 3.1.3 Bootstrap Loaders

These routines provide mechanisms to load a user program via the selected interface by moving code into the Program Scratchpad RAM of CPU0 (CPU0\_PSPR). The loaded code is started after exiting the BootROM.

**Note:** Once a Bootstrap Loader mode is entered, the selected communication protocol (CAN/ASC) must be completely executed according to its definition (as of the below Sections) until the user code is downloaded into the device. No time-out will ever interrupt this process; only a reset can re-start the device.

## AURIX™ TC3xx Platform Firmware

**Table 55 HW Configuration Data for Bootstrap Loader Modes**

Bootstrap Loader Mode	Channel/node	RxD Line	TxD Line
ASC Bootstrap Loader mode	ASCLIN0	P15.3	P15.2
Generic Bootstrap Loader mode - ASC protocol	ASCLIN0	P14.1	P14.0
Generic Bootstrap Loader mode - CAN protocol	MCMCAN0 module CAN1 node	P14.1	P14.0

### 3.1.3.1 ASC Bootstrap loader

The ASC Bootloading routine implements the following steps:

- RxD/TxD pin configuration is done in accordance with the AURIX™ TC3xx Platform definitions. Depending on the start-up mode, the routine is invoked in “ASC Bootloader”-startup mode (ASC-only pins are used), or if ASC protocol is detected in “Generic Bootloader”-mode (CAN/ASC-shared pins are used but configured to ASC module)
- baudrate calculation is done based on the zero Byte sent by the host
- ASCLIN is initialized (without enabling the receiver) to the baudrate as determined (8 data and 1 stop bit)
- acknowledge byte D5H is sent to the host, indicating the device is ready to accept a data transfer
- after the acknowledge byte is transmitted, the receiver is enabled
- the bootloader enters a loop, waiting to receive exactly 128 bytes which are stored as 32 words in CPU0 Program Scratchpad RAM starting from address C000 0000H

Once 128 bytes are received, the SSW continues further - refer to [Figure 22](#). After exiting the SSW, user code will be started from address C000 0000H (CPU0\_PSPR).

Note: *ASC Bootstrap Loader supports also half-duplex mode but this does not mean the RxD and TxD pins should be in such a case directly short-connected at the device boundary. The “single transmission line” must be supported otherwise - the typical implementation includes CAN transceiver circuits respectively using CAN physical layer for data transfer.*

## AURIX™ TC3xx Platform Firmware

### 3.1.3.2 CAN Bootstrap loader

The CAN bootstrap loader (CAN BSL) transfers program code/data from an external host via CAN interface into AURIX™ TC3xx Platform CPU0 Program Scratchpad RAM. It is a primary bootloader, not to be mixed up with the secondary bootloader which can be bought from tool partners.

#### 3.1.3.2.1 CAN BSL summary

The main characteristics of AURIX™ TC3xx Platform CAN Bootstrap Loader include:

- Classical CAN is supported as well as CAN FD (CAN FD only in case of [Configured OSC](#))
- the initial identifier is fixed to 555H
- the number of data frames to be received by BSL is programmable
- for other CAN BSL characteristics - refer to [Chapter 3.1.3.2.4](#)

The communication between the AURIX™ TC3xx Platform and an external host is based on the following three CAN standard frames:

- Initialization frame(s) - sent by the external host to AURIX™ TC3xx Platform
- Acknowledgement frame(s) - sent by the AURIX™ TC3xx Platform to the external host
- Data frame(s) - sent by the external host to AURIX™ TC3xx Platform

#### 3.1.3.2.2 Clock system during CAN BSL

There are two scenarios of AURIX™ TC3xx Platform CAN BSL usage in regard to the clock system, depending on the Oscillator Circuit OSC configuration (refer to “Clocking System” Chapter), whereas the following basics apply:

- the OSC configuration is primary controlled by DMU\_HF\_PROCONDF.OSCCFG bit
- the OSCCFG bit as the complete DMU\_HF\_PROCONDF register is installed during device start-up from UCB\_DFLASH (refer to the “User Configuration Block (UCB)” and the “Data Memory Unit (DMU)” Sections in the “Non Volatile Memory (NVM) Subsystem” Chapter)
- UCB\_DFLASH is user-programmable

The two scenarios for clocking during CAN BSL are described below.

#### Not configured OSC

This is the clock system status after device system and power-on reset if DMU\_HF\_PROCONDF.OSCCFG=0 and usually it's on place in delivery state of the device - meaning if UCB\_DFLASH has not been re-programmed by the customer with OSC information as proper for the design/board where the chip will be used.

In such a case the back-up clock (refer to “Back-up Clock” Section in “Clocking System” Chapter) is used as clock-source for MCAN module during CAN BSL, after performing the following configurations:

- fBACK/5 is selected as clock source fMCANI - CCU\_CCUCON1.MCANDIV=5
- fMCANI is selected as clock source fMCAN - CCU\_CCUCON1.CLKSELMCAN=01B

Following from the above, the MCAN module operates with 20MHz clock for the asynchronous part as provided by the AURIX™ TC3xx Platform internal back-up clock source. In this scenario, less baudrates are supported by CAN BSL (refer to the next Chapter).

#### Configured OSC

OSC configuration is performed by device start-up procedure if DMU\_HF\_PROCONDF.OSCCFG=1 installing information into OSC Control register CCU\_OSCCON according to the relevant UCB\_DFLASH resp. DMU\_HF\_PROCONDF content (refer to “Configuration of the Oscillator” Section in “Clocking System” Chapter).

## AURIX™ TC3xx Platform Firmware

To come into this state and to use fully the CAN BSL functionality, the customer must first re-program the PROCONDF location (related bits/fields) inside UCB\_DFLASH in accordance to the design/board where the chip will be used.

In such a case, the OSC is activated and operating on the frequency provided by the external source (crystal/resonator) connected to XTAL1/2 pins (refer to “External Crystal / Ceramic Resonator Mode” Section in “Clocking System” Chapter), with configuration performed during start-up:

- FOSC0 is selected as clock source fMCAN - CCU\_CCUCON1.CLKSELMCAN=10B

Following from the above, the MCAN module operates with clock for the asynchronous part as provided by the OSC circuit driven by the external source at XTAL, so supporting the full range of baudrates according to [Table 56](#).

### 3.1.3.2.3 CAN BSL usage after application reset

The following scenario:

- the device has been started in CAN BSL mode with a power-on or system reset, AND
- CAN BSL must be executed again upon the consequent application reset(s)

is not feasible for devices with not configured OSC.

If such use-case must be supported - then first the PROCONDF location (related bits/fields) inside UCB\_DFLASH need to be re-programmed to configure the OSC.

### 3.1.3.2.4 Supported CAN features

The number of supported Baud rates and CAN operational modes, as well as the sampling point to be configured on the host side depend on the clock system status during CAN BSL (refer to [Chapter 3.1.3.2.2](#)).

#### Not configured OSC - back-up clock used for CAN

When working with the internal back-up clock source, AURIX™ TC3xx Platform CAN BSL supports:

- Classical CAN only
- sample point of 60%
- limited number of baudrates:
  - 100 KBit/s
  - 125 KBit/s
  - 250 KBit/s
  - 500 KBit/s

#### Configured OSC - external clock source used for CAN

When working with the external clock source at XTAL, AURIX™ TC3xx Platform CAN BSL supports:

- Classical CAN and CAN FD
- sample point of 80%
- a big number of combinations between Classical CAN Baud rates and XTAL/FOSC frequency as shown in [Table 56](#), whereas:
  - “X” marks supported combination
  - “-” marks not supported combination

## AURIX™ TC3xx Platform Firmware

**Table 56 Possible Baudrate and FOSC frequency combinations**

Baudrate (KBits/s) / FOSC (MHz)	4	5	8	10	12	16	20	24	25	40
20	X	X	X	X	X	X	X	X	X	X
25	X	X	X	X	X	X	X	X	X	X
30	-	-	-	-	X	-	-	X	-	-
33	X	X	X	X	X	X	X	X	X	X
40	X	-	X	X	X	X	X	X	-	X
50	X	X	X	X	X	X	X	X	X	X
60	-	-	-	-	X	-	-	X	-	-
66	X	X	X	X	X	X	X	X	X	X
80	X	-	X	-	X	X	X	X	-	X
100	X	X	X	X	X	X	X	X	X	X
125	X	X	X	X	X	X	X	X	X	X
133	X	-	X	X	X	X	X	X	-	X
150	-	-	-	-	X	-	-	X	-	-
166	X	X	X	X	X	X	X	X	X	X
200	X	-	X	X	X	X	X	X	-	X
250	X	X	X	X	X	X	X	X	X	X
266	X	-	X	-	X	X	X	X	-	X
300	-	-	-	-	X	-	-	X	-	-
333	X	X	X	X	X	X	X	X	X	X
375	-	-	-	-	X	-	-	X	-	-
400	X	-	X	-	X	X	X	X	-	X
500	X	X	X	X	X	X	X	X	X	X
533	-	-	X	-	-	X	-	X	-	X
600	-	-	-	-	X	-	-	X	-	-
625	-	X	-	X	-	-	-	-	X	X
666	-	-	X	X	X	X	X	X	-	X
750	-	-	-	-	X	-	-	X	-	-
800	-	-	X	-	X	X	-	X	-	X
833	-	-	-	X	-	-	X	-	X	X
1000	-	-	X	X	X	X	X	X	-	X

**3.1.3.2.5 CAN BSL flow**

The CAN bootstrap loader flow is shown on [Figure 24](#) and it consists of 3 phases which are described in the next Sections.

The notations used in the below descriptions are:

- ACKID - Acknowledgement Message shall have this 11-bit Identifier (like in register)

## AURIX™ TC3xx Platform Firmware

- DMSGC - Amount of messages, which will be sent including the program
- DMSGID - The data frames will have this 11-bit ID (like in register)
- TSEG1 and TSEG2 - Baudrate as used (copied from register)
- NBTP - Nominal bit timing and Prescaler Register Settings (Classical CAN has arbitration segment only)
- DBTP - Data bit timing and Prescaler Register Settings (Fast Segment)
- DBPM - Data Bytes per Message (DLC= [9, 11] is not allowed)

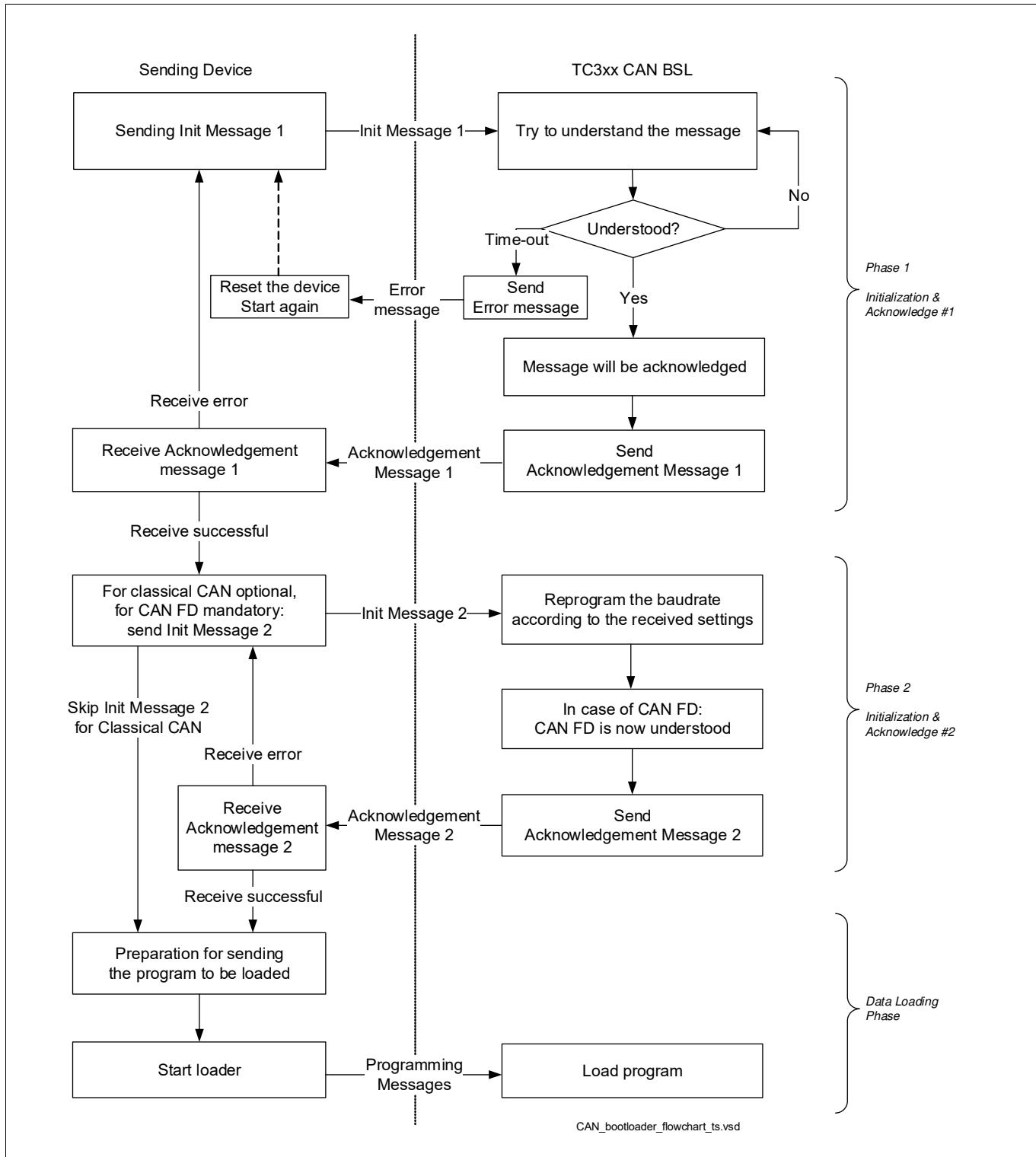


Figure 24 CAN Bootstrap loader flow

## AURIX™ TC3xx Platform Firmware

### Phase 1 (Initialization & Acknowledge #1)

During this phase, CAN BSL tries to determine the CAN baud rate at which the external host is communicating and to understand (receive correctly) the initialization message (classical CAN message) sent by the host.

The above requires that external host sends continuously to the AURIX™ TC3xx Platform **Initialization Message 1** in classical CAN format as shown in **Table 57**.

**Table 57 Initialization Message 1**

Frame Format requested	ID	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
Classical CAN	0x555	8	0x55	0x55	ACKID Low	ACKID High	DMSGC Low	DMSGC High	DMSGID Low	DMSGID High
CAN FD	0x555	10	0x55	0x55	ACKID Low	ACKID High	DMSGC Low	DMSGC High	DMSGID Low	DMSGID High

CAN BSL evaluates the incoming messages trying to identify a content according to **Table 57** within the range of baudrate configurations from **Table 56** and according to the result:

- if the message is understood - then the CAN BSL
  - configures the MCAN module for the selected baudrate settings
  - acknowledges the next received **Initialization Message 1** - Tx ACK bit field as dominant
  - sends **Acknowledgement Message 1** with the format shown in **Table 58**; this message also informs the host whether the reception was error-free or not
- if initialization messages can not be understood for any supported baudrate settings
  - CAN BSL sends Error message with the format shown in **Table 59**
  - CAN BSL terminates - the device should be re-started with a reset

**Table 58 Acknowledgement Message 1**

Frame	ID	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
Message received error free	ACKID	8 or 10	ACKID Low	ACKID High	DMSGC Low	DMSGC High	DMSGID Low	DMSGID High	TSEG1	TSEG2
Faulty DLC	ACKID or if not received 0xAA	as received	0xAA	0xAA	0xAA	0xAA	0xAA	0xAA	0xAA	0xAA

**Table 59 Error message (No baudrate detected)**

Frame	ID	DLC
I don't understand	0xAAA	0

### Phase 2 (Initialization & Acknowledge #2)

Once having **Acknowledgement Message 1** from CAN BSL successfully received, the host can send **Initialization Message 2** with the format shown in **Table 60**, whereas this message is:

- required - if CAN FD must be used further on

## AURIX™ TC3xx Platform Firmware

- optional - allowing the host to provide a more precise baud rate for further classical CAN operation; in such a case, jump in baudrate in any direction for the arbitration segment is not allowed

**Table 60 Initialization Message 2**

Frame Format requested	ID	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
Classical CAN	0x555	4	NBTP. NTSEG2	NBTP. NTSEG1	NBTP. [23..16]	NBTP. [31..24]				
CAN FD	0x555	8	NBTP. NTSEG2	NBTP. NTSEG1	NBTP. [23..16]	NBTP. [31..24]	DBTP.[ 7..0]	DBTP.[ 15..8]	DBTP.[ 23..16]	DBPM

If after sending **Acknowledgement Message 1**, **Initialization Message 2** is received but not **Programming message** (differentiation done upon the message ID) - the CAN BSL analyses this message against the format shown in **Table 60** and then:

- if classical CAN has been requested (DLC=8)
  - install the (more precise) baudrate settings (NBTP register) as received
  - send **Acknowledgement Message 2** (refer to **Table 60**)
- if CAN FD has been requested (DLC=4)
  - enable CAN FD
  - install the baudrate settings (NBTP and DBTP registers) as received
  - take the received DBPM to be used as DLC for the upcoming Programming messages
  - send **Acknowledgement Message 2** (refer to **Table 60**)
- if the message has been wrongly received - DLC other than 4 or 8)
  - send **Acknowledgement Message 2** indicating “Faulty DLC” (refer to **Table 60**)

**Table 61 Acknowledgement Message 2**

Frame	ID	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
Message received error free	ACKID	4 or 8	NBTP.N TSEG2	NBTP.N TSEG1	NBTP. [23..16]	NBTP. [31..24]	DBTP. [7..0] (CAN FD only)	DBTP. [15..8] (CAN FD only)	DBTP. [23..16] (CAN FD only)	DBPM (CAN FD only)
Faulty DLC	ACKID	as received	0xAA	0xAA	0xAA	0xAA	0xAA	0xAA	0xAA	0xAA

**Acknowledgement Message 2** sent by CAN BSL and indicating “Error free” tells the external host, the device is now ready to receive data frames.

### Data Loading Phase

After receiving **Acknowledgement Message 2** from CAN BSL or if **Phase 2 (Initialization & Acknowledge #2)** has been skipped, the host starts sending **Programming messages** with the format shown in **Table 62**. CAN BSL receives the data and stores it CPU0 Program Scratchpad RAM starting from address C000 0000H.

Both communication partners evaluate the data message count until the requested number of CAN data frames has been transmitted.

After the reception of the last CAN data frame, CAN BSL terminates and returns to the main SSW flow (refer to **Figure 22**).

## AURIX™ TC3xx Platform Firmware

After exiting the SSW, user code will be started from address C000 0000H (CPU0 PSPR).

**Table 62 Programming message**

Frame Format requested	ID	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
Classical CAN	DMSGID as ID not as register entry	8	Hex Code Byte	Hex Code Byte	Hex Code Byte	Hex Code Byte	Hex Code Byte	Hex Code Byte	Hex Code Byte	Hex Code Byte
CAN FD	DMSGID as ID not as registry entry	DBPM	Hex Code Bytes amount of Data Bytes is defined in DBPM							

### 3.1.4 Support for Software over the Air (SOTA)

After power-on and system reset, the AURIX™ TC3xx Platform SSW evaluates the SOTA configuration installed in UCB\_SWAP\_ORIG/COPY and enables accordingly the SWAP functionality which allows to switch between the banks of the device PFLASH.

If a valid SOTA configuration is found in UCB\_SWAP\_ORIG/COPY, then the SSW executes the following:

- disable the direct access to PFLASH (the direct read paths CPUx-PFLx) by installing 1 into all CPUx\_FLASHCON4.DDIS bits
- install into SRU\_SWAPCTRL.ADDRCFG register:
  - Address region A active - if SWAP\_A marker is found
  - Address region B active - if SWAP\_B marker is found

### 3.1.5 Shutdown request handler

All the active CPUs in AURIX™ TC3xx Platform jump unconditionally to the entry point of this handler upon any warm reset request.

Hardware guarantees that this handler can not be interrupted by any other (interrupt/trap) request. Once the handler ends, all the CPUs are in stable passive state reached by a controlled ramp-down sequence, preventing big current jumps.

At its entry point (common for all the CPUs), the firmware causes any running CPU to jump to its own handler. For this purpose, the CORE\_ID register is read; because this register value is individual for any CPU - upon CORE\_ID=0, 1, 2, 3, 4 or 6 the firmware jumps to the routine for the respective CPU.

Note: for CPU5, CORE\_ID=6.

The functionality of all handlers is similar, namely:

- prepare work data for execution of average-power loop
- execute average-power loop until SCU\_RSTCON2.TOUTyy=1
- execute WAIT instruction

## AURIX™ TC3xx Platform Firmware

Upon completion of the above sequence, CPUx reaches a passive state yy microseconds after shutdown request activation, where:

- yy=20 for CPU2 and CPU5
- yy=40 for CPU1 and CPU4
- yy=60 for CPU0 and CPU3

### 3.1.6 Power Supply Friendly Debug Monitor

The AURIX™ TC3xx Platform BootROM contains a routine called Power Supply Friendly Debug Monitor (PSFDM). The purpose of this routine is to minimize the risk of getting EVR voltage over/undershoot due to a sudden current drop when more than one CPU is halted by OCDS, and to avoid a current peak when the CPUs are released from halt.

The PSFDM routine is an independent/stand alone module inside the BootROM that is not executed during device start-up. It is intended to be used by CPUs as a debug trap handler instead of halting. This means the debugger must configure properly the debug trap vector - it should point to address AFFF FC80H (inside BootROM).

Upon a debug event, the trap will be triggered for all configured CPUs, starting PSFDM execution. The CPUs can be halted individually by the tool if needed.

The PSFDM routine contains a “repeat action until condition” loop:

- the action represents a sequence of instructions that is meant to consume as much power as a typical application. For this purpose, a mixture between two TriCore instructions is implemented:
  - most power intensive (MADD.Q)
  - least power intensive (NOP)
- the condition to exit the loop is CBS\_TLS.TL2=0 (OTGS Trigger Line 2 is deactivated)

To continue user code execution, the debugger must:

- release from halt those CPUs which have been halted - so all the CPUs are running PSFDM as debug trap handler
- activate OTGS Trigger Line 2 - all the CPUs exit PSFDM (by RTE) and continue user code execution

At the end, after debug trap the CPUs are restarted in parallel, with a few cycles slack due to the individual CBS\_TLS.TL2 polling.

## 3.2 Registers

AURIX™ TC3xx Platform contains several registers exclusively dedicated to usage by Firmware as below described. These registers provide user with information about the results from start-up mode evaluation done by SSW.

## AURIX™ TC3xx Platform Firmware

## 3.2.1 Firmware specific usage of device registers

## 3.2.1.1 Registers providing information on the boot selections

## SCU\_STMEM1

## Start-up Memory Register 1

(for address - refer to SCU Chapter)

Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES								SWAP_DW_INDEX							
r					r			r			r	r	r	r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	HARREQ	RES		BOOTMODE_CONFIGURED	SCR_DIS	BOOT_CFG		BMHD_INDEX		BMHD_COPY	BOOT_PIN	BMI_VALID	1		
r	r	r		r	r	r		r		r	r	r	r	r	r

Field	Bits	Type	Description
1	0:0	r	<b>Reserved</b>
BMI_VALID	1:1	r	<b>BMI valid flag</b> 0 <sub>B</sub> no valid BMHD/BMI found by the last SSW execution 1 <sub>B</sub> valid BMHD/BMI was found by the last SSW execution
BOOT_PIN	2:2	r	<b>Configuration from pins flag<sup>1)</sup></b> 0 <sub>B</sub> HWCFG pins were not checked during the last SSW execution 1 <sub>B</sub> start-up mode selected by HWCFG pins during the last SSW execution
BMHD_COPY	3:3	r	<b>Valid BMHD Copy flag<sup>1)</sup></b> 0 <sub>B</sub> start-up mode selection done based on BMHD[BMHD_INDEX] Original value (from BMHDx_ORIG_UCB) 1 <sub>B</sub> start-up mode selection done based on BMHD[BMHD_INDEX] Copy value (from BMHDx_COPY_UCB)
BMHD_INDEX	5:4	r	<b>Index of the valid BMHD<sup>1)</sup></b> 00 <sub>B</sub> start-up mode selection done based on BMHD0 01 <sub>B</sub> start-up mode selection done based on BMHD1 10 <sub>B</sub> start-up mode selection done based on BMHD2 11 <sub>B</sub> start-up mode selection done based on BMHD3
BOOT_CFG	8:6	r	<b>Start-up mode effectively taken by SSW<sup>2)</sup></b> 111 <sub>B</sub> Internal start from Flash 110 <sub>B</sub> Alternate Boot Mode (ABM, Generic BSL on fail) 101 <sub>B</sub> Alternate Boot Mode (ABM, ASC BSL on fail) 100 <sub>B</sub> Generic Bootstrap Loader Mode (ASC/CAN BSL) 011 <sub>B</sub> ASC Bootstrap Loader Mode (ASC BSL)
SCRDIS	9:9	r	<b>SCR disabled flag<sup>3)</sup></b> 0 <sub>B</sub> SCR is not disabled by SSW 1 <sub>B</sub> SCR is disabled by SSW due to start-up failure

## AURIX™ TC3xx Platform Firmware

Field	Bits	Type	Description
<b>BOOTMODE_CONFIGURED</b>	10:10	r	<b>Boot Mode Configured flag</b> 0 <sub>B</sub> NO start-up (boot) mode has been selected based on BMHDx/pins by the last SSW execution 1 <sub>B</sub> start-up (boot) mode has been selected based on BMHDx/pins by the last SSW execution
<b>RES</b>	13:11	r	<b>Reserved, can show any value</b>
<b>HARREQ</b>	14:14	r	<b>Halt-After-Reset REQuest flag</b> 0 <sub>B</sub> no halt-after-reset requested during the last SSW execution 1 <sub>B</sub> halt-after-reset request received during the last SSW execution
<b>RES</b>	15:15	r	<b>Reserved, can show any value</b>
<b>SWAP_CFG</b>	17:16	r	<b>SWAP configuration</b> 00 <sub>B</sub> no SWAP configured by SSW (full PFlash address space active with the default map) 01 <sub>B</sub> SWAP A configured (PFlash Bank A active, B inactive) 10 <sub>B</sub> SWAP B configured (PFlash Bank B active, A inactive) 11 <sub>B</sub> reserved
<b>SWAP_TARGET</b>	18:18	r	<b>UCB_SWAP used for configuration flag</b> <sup>4)</sup> 0 <sub>B</sub> SWAP configuration done based on UCB_SWAP_ORIG 1 <sub>B</sub> SWAP configuration done based on UCB_SWAP_COPY
<b>SWAP_DW_INDEX</b>	23:19	r	<b>Offset from UCB_SWAP_ORIG/COPY begin</b> <sup>4)</sup> <b>of the double-word aligned location where from the SWAP configuration (A/B) has been installed</b>
<b>RES</b>	31:24	r	<b>Reserved, can show any value</b>

1) Value to be considered as valid only if BMI\_VALID=1

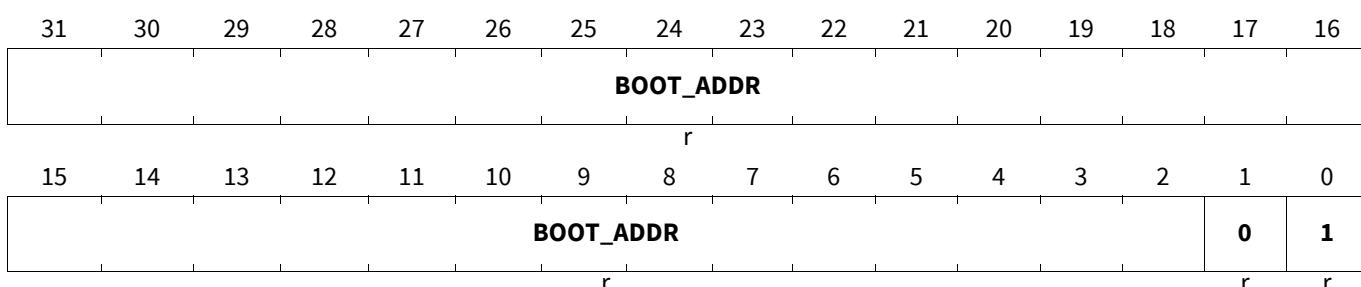
2) The value here can be different from the BMI from the valid BMHD - e.g. if configuration from pins is taken

3) The user software must take into account, that SCR is started by SSW upon cold power-on only, but SCRDIS flag will be cleared by the next warm power-on

4) Value to be considered as valid only if SWAP\_CFG=[01b, 10b]

**SCU\_STMEM2****Start-up Memory Register 2**

(for address - refer to SCU Chapter)

**Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>1</b>	0:0	r	<b>Reserved</b>
<b>0</b>	1:1	r	<b>Reserved</b>

## AURIX™ TC3xx Platform Firmware

Field	Bits	Type	Description
BOOT_ADDR	31:2	r	<b>Address of the first user-code instruction, taken after SSW</b> <b>NOTE:</b> this address is always word-aligned, therefore bits [1:0] are effectively taken as zero

**3.2.1.2 Registers providing information on the Checker Software activity**

Note: *EMEM-related checks - indicated by STMEMx[17:16] - are only performed on devices implementing Emulation Memory (EMEM) and only for those EMEM module(s) which are effectively present on the product.*

*The proper handling on this topic is assured by respective CHSW Reference Tables content in Config Sector (CFS).*

**SCU\_STMEM3**

**Start-up Memory Register 3 (for address - refer to SCU Chapter)**      **Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FM TRIM _CS	0	FLASH CON1 _CS		0			GETH 1 _CS	RIF1 _CS	RIFO _CS	GETH _CS	CONV CTRL _CS	0		EMEM T2 _CS	EMEM T1 _CS
r	r	r		r			r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWAIT _CS	PLL TRIM _CS	CCU CON0 _CS	CHIP1 D _CS	RST CON3 _CS	0	EVRT2 _CS	EVRT1 _CS	APPR _CS	SYSR _CS	WPOR _CS	CPOR _CS	SWAP _CS	BMSEL _CS	CHSW GEN _CS	1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
1	0:0	r	<b>Reserved</b>
CHSWGEN_CS	1:1	rw	<b>CHSW general status</b> 0 <sub>B</sub> CHSW not started 1 <sub>B</sub> CHSW started
BMSEL_CS	2:2	rw	<b>Bootmode selection check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
SWAP_CS	3:3	rw	<b>SWAP configuration check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
CPOR_CS	4:4	rw	<b>Reset type evaluation check upon cold power-on</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
WPOR_CS	5:5	rw	<b>Reset type evaluation check upon warm power-on</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started

## AURIX™ TC3xx Platform Firmware

Field	Bits	Type	Description
<b>SYSR_CS</b>	6:6	rw	<b>Reset type evaluation check upon system reset</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
<b>APPR_CS</b>	7:7	rw	<b>Reset type evaluation check upon application reset</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
<b>EVRT1_CS</b>	8:8	rw	<b>EVR &amp; power-system register registers check #1</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
<b>EVRT2_CS</b>	9:9	rw	<b>EVR &amp; power-system register registers check #2</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
<b>RSTCON3_CS</b>	11:11	rw	<b>RCU_RSTCON3 register check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
<b>CHIPID_CS</b>	12:12	rw	<b>SCU_CHIPID register check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
<b>CCUCONO_CS</b>	13:13	rw	<b>CCU_CCUCONO register check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
<b>PLLTRIM_CS</b>	14:14	rw	<b>System &amp; peripheral PLLs trimming check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
<b>PWAIT_CS</b>	15:15	rw	<b>PFLASH wait states check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
<b>EMEMT1_CS</b>	16:16	rw	<b>MC_ECCS register check for EMEM0 and EMEM3</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
<b>EMEMT2_CS</b>	17:17	rw	<b>MEMCON register check for EMEM0 and EMEM3</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
<b>0</b>	19:18, 10	rw	<b>Reserved</b>
<b>CONVCTRL_CS</b>	20:20	rw	<b>Converter control block trimming check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
<b>GETH_CS</b>	21:21	rw	<b>Gigabit Ethernet MAC module calibration check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
<b>RIFO_CS</b>	22:22	rw	<b>RIFO module calibration check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started

## AURIX™ TC3xx Platform Firmware

Field	Bits	Type	Description
RIF1_CS	23:23	rw	<b>RIF1 module calibration check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
GETH1_CS	24:24	rw	<b>Gigabit Ethernet MAC second module calibration check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
0	30,28:25	rw	<b>Reserved</b>
FLASHCON1_CS	29:29	rw	<b>CPUx_FLASHCON1 registers check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started
FMTRIM_CS	31:31	rw	<b>FM amplitude trimming check</b> 0 <sub>B</sub> check not started 1 <sub>B</sub> check started

## SCU\_STMEM4

Start-up Memory Register 4

(for address - refer to SCU Chapter)

Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FM TRIM _CF	0	FLASH CON1 _CF		0		GETH 1 _CF	RIF1 _CF	RIFO _CF	GETH _CF	CONV CTRL _CF		0		EMEM T2 _CF	EMEM T1 _CF
r	r	r		r		r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWAIT _CF	PLL TRIM _CF	CCU CON0 _CF	CHIPID _CF	RST CON3 _CF	0	EVRT2 _CF	EVRT1 _CF	APPR _CF	SYSR _CF	WPOR _CF	CPOR _CF	SWAP _CF	BMSEL _CF	CHSW GEN _CF	1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
1	0:0	r	<b>Reserved</b>
CHSWGEN_CF	1:1	rw	<b>CHSW general status</b> 0 <sub>B</sub> CHSW not executed or passed 1 <sub>B</sub> CHSW failed
BMSEL_CF	2:2	rw	<b>Bootmode selection check</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
SWAP_CF	3:3	rw	<b>SWAP configuration check</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
CPOR_CF	4:4	rw	<b>Reset type evaluation check upon cold power-on</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed

## AURIX™ TC3xx Platform Firmware

Field	Bits	Type	Description
WPOR_CF	5:5	rw	<b>Reset type evaluation check upon warm power-on</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
SYSR_CF	6:6	rw	<b>Reset type evaluation check upon system reset</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
APPR_CF	7:7	rw	<b>Reset type evaluation check upon application reset</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
EVRT1_CF	8:8	rw	<b>EVR &amp; power-system register registers check #1</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
EVRT2_CF	9:9	rw	<b>EVR &amp; power-system register registers check #2</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
RSTCON3_CF	11:11	rw	<b>RCU_RSTCON3 register check</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
CHIPID_CF	12:12	rw	<b>SCU_CHIPID register check</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
CCUCON0_CF	13:13	rw	<b>CCU_CCUCON0 register check</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
PLLTRIM_CF	14:14	rw	<b>System &amp; peripheral PLLs trimming check</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
PWAIT_CF	15:15	rw	<b>PFLASH wait states check</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
EMEMT1_CF	16:16	rw	<b>MC_ECCS register check for EMEM0 and EMEM3</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
EMEMT2_CF	17:17	rw	<b>MEMCON register check for EMEM0 and EMEM3</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
0	19:18, 10	rw	<b>Reserved</b>
CONVCTRL_CF	20:20	rw	<b>Converter control block trimming check</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
GETH_CF	21:21	rw	<b>Gigabit Ethernet MAC module calibration check<sup>1)</sup></b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed

## AURIX™ TC3xx Platform Firmware

Field	Bits	Type	Description
RIF0_CF	22:22	rw	<b>RIF0 module calibration check</b> <sup>2)</sup> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
RIF1_CF	23:23	rw	<b>RIF1 module calibration check</b> <sup>2)</sup> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
GETH1_CF	24:24	rw	<b>Gigabit Ethernet MAC second module calibration check</b> <sup>1)</sup> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
0	30,28:25	rw	<b>Reserved</b>
FLASHCON1_CF	29:29	rw	<b>CPUx_FLASHCON1 registers check</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed
FMTRIM_CF	31:31	rw	<b>FM amplitude trimming check</b> 0 <sub>B</sub> check not executed or passed 1 <sub>B</sub> check failed

- 1) The check for Gigabit Ethernet MAC module(s) calibration will fail after application reset, if the application software has not enabled GETH clock (in SCU\_CCUCON5.GETHDIV register) after the previous power-on/system reset(s) - meaning if the module is not used by application - therefore in such use-case anyway the check for this' module calibration is not relevant.
- 2) The check for RIF module(s) calibration will fail after application reset, if the application software has not enabled ADAS clock (in SCU\_CCUCON.ADASDIV register) after the previous power-on/system reset(s) - meaning if the module is not used by application - therefore in such use-case anyway the check for this' module calibration is not relevant.

## SCU\_STMEM5

## Start-up Memory Register 5

(for address - refer to SCU Chapter)

Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FM TRIM _CP	0	FLASH CON1 _CP		0		GETH 1 _CP	RIF1 _CP	RIFO _CP	GETH _CP	CONV CTRL _CP		0		EMEM T2 _CP	EMEM T1 _CP
r	r	r		r		r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWAIT _CP	PLL TRIM _CP	CCU CON0 _CP	CHIPID _CP	RST CON3 _CP	0	EVRT2 _CP	EVRT1 _CP	APPR _CP	SYSR _CP	WPOR _CP	CPOR _CP	SWAP _CP	BMSEL _CP	CHSW GEN _CP	1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
1	0:0	r	<b>Reserved</b>
CHSWGEN_CP	1:1	rw	<b>CHSW general status</b> 0 <sub>B</sub> CHSW not executed or failed 1 <sub>B</sub> CHSW passed

## AURIX™ TC3xx Platform Firmware

Field	Bits	Type	Description
<b>BMSEL_CP</b>	2:2	rw	<b>Bootmode selection check</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>SWAP_CP</b>	3:3	rw	<b>SWAP configuration check</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>CPOR_CP</b>	4:4	rw	<b>Reset type evaluation check upon cold power-on</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>WPOR_CP</b>	5:5	rw	<b>Reset type evaluation check upon warm power-on</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>SYSR_CP</b>	6:6	rw	<b>Reset type evaluation check upon system reset</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>APPR_CP</b>	7:7	rw	<b>Reset type evaluation check upon application reset</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>EVRT1_CP</b>	8:8	rw	<b>EVR &amp; power-system register registers check #1</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>EVRT2_CP</b>	9:9	rw	<b>EVR &amp; power-system register registers check #2</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>RSTCON3_CP</b>	11:11	rw	<b>RCU_RSTCON3 register check</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>CHIPID_CP</b>	12:12	rw	<b>SCU_CHIPID register check</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>CCUCON0_CP</b>	13:13	rw	<b>CCU_CCUCON0 register check</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>PLLTRIM_CP</b>	14:14	rw	<b>System &amp; peripheral PLLs trimming check</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>PWAIT_CP</b>	15:15	rw	<b>PFLASH wait states check</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>EMEMT1_CP</b>	16:16	rw	<b>MC_ECCS register check for EMEM0 and EMEM3</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
<b>EMEMT2_CP</b>	17:17	rw	<b>MEMCON register check for EMEM0 and EMEM3</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed

## AURIX™ TC3xx Platform Firmware

Field	Bits	Type	Description
0	19:18, 10	rw	<b>Reserved</b>
CONVCTRL_CP	20:20	rw	<b>Converter control block trimming check</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
GETH_CP	21:21	rw	<b>Gigabit Ethernet MAC module calibration check</b> <sup>1)</sup> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
RIF0_CP	22:22	rw	<b>RIF0 module calibration check</b> <sup>2)</sup> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
RIF1_CP	23:23	rw	<b>RIF1 module calibration check</b> <sup>2)</sup> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
GETH1_CP	24:24	rw	<b>Gigabit Ethernet MAC second module calibration check</b> <sup>1)</sup> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
0	30,28:25	rw	<b>Reserved</b>
FLASHCON1_CP	29:29	rw	<b>CPUX_FLASHCON1 registers check</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed
FMTRIM--_CP	31:31	rw	<b>FM amplitude trimming check</b> 0 <sub>B</sub> check not executed or failed 1 <sub>B</sub> check passed

- 1) The check for Gigabit Ethernet MAC module calibration will fail after application reset, if the application software has not enabled GETH clock (in SCU\_CCUCON5.GETHDIV register) after the previous power-on/system reset(s) - meaning if the module is not used by application - therefore in such use-case anyway the check for this' module calibration is not relevant.
- 2) The check for RIF module(s) calibration will fail after application reset, if the application software has not enabled ADAS clock (in SCU\_CCUCON5.ADASDIV register) after the previous power-on/system reset(s) - meaning if the module is not used by application - therefore in such use-case anyway the check for this' module calibration is not relevant.

## SCU\_STMEM6

## Start-up Memory Register 6

(for address - refer to SCU Chapter)

Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FM TRIM _CE	0	FLASH CON1 _CE		0		GETH 1 _CE	RIF1 _CE	RIFO _CE	GETH _CE	CONV CTRL _CE	0	EMEM T2 _CE	EMEM T1 _CE		
r	r	r		r		r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWAIT _CE	PLL TRIM _CE	CCU CON0 _CE	CHIPID _CE	RST CON3 _CE	0	EVRT2 _CE	EVRT1 _CE	APPR _CE	SYSR _CE	WPOR _CE	CPOR _CE	SWAP _CE	BMSEL _CE	CHSW GEN _CE	1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

## AURIX™ TC3xx Platform Firmware

Field	Bits	Type	Description
<b>1</b>	0:0	r	<b>Reserved</b>
<b>CHSWGEN_CE</b>	1:1	rw	<b>CHSW general status</b> 0 <sub>B</sub> CHSW not executed 1 <sub>B</sub> CHSW finished
<b>BMSEL_CE</b>	2:2	rw	<b>Bootmode selection check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>SWAP_CE</b>	3:3	rw	<b>SWAP configuration check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>CPOR_CE</b>	4:4	rw	<b>Reset type evaluation check upon cold power-on</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>WPOR_CE</b>	5:5	rw	<b>Reset type evaluation check upon warm power-on</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>SYSR_CE</b>	6:6	rw	<b>Reset type evaluation check upon system reset</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>APPR_CE</b>	7:7	rw	<b>Reset type evaluation check upon application reset</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>EVRT1_CE</b>	8:8	rw	<b>EVR &amp; power-system register registers check #1</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>EVRT2_CE</b>	9:9	rw	<b>EVR &amp; power-system register registers check #2</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>RSTCON3_CE</b>	11:11	rw	<b>RCU_RSTCON3 register check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>CHIPID_CE</b>	12:12	rw	<b>SCU_CHIPID register check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>CCUCONO_CE</b>	13:13	rw	<b>CCU_CCUCONO register check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>PLLTRIM_CE</b>	14:14	rw	<b>System &amp; peripheral PLLs trimming check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>PWAIT_CE</b>	15:15	rw	<b>PFLASH wait states check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished

## AURIX™ TC3xx Platform Firmware

Field	Bits	Type	Description
<b>EMEMT1_CE</b>	16:16	rw	<b>MC_ECCS register check for EMEM0 and EMEM3</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>EMEMT2_CE</b>	17:17	rw	<b>MEMCON register check for EMEM0 and EMEM3</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>0</b>	19:18, 10	rw	<b>Reserved</b>
<b>CONVCTRL_CE</b>	20:20	rw	<b>Converter control block trimming check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>GETH_CE</b>	21:21	rw	<b>Gigabit Ethernet MAC module calibration check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>RIF0_CE</b>	22:22	rw	<b>RIF0 module calibration check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>RIF1_CE</b>	23:23	rw	<b>RIF1 module calibration check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>GETH1_CE</b>	24:24	rw	<b>Gigabit Ethernet MAC second module calibration check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>0</b>	30,28:25	rw	<b>Reserved</b>
<b>FLASHCON1_CE</b>	29:29	rw	<b>CPUx_FLASHCON1 registers check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished
<b>FMTRIM_CE</b>	31:31	rw	<b>FM amplitude trimming check</b> 0 <sub>B</sub> check not executed 1 <sub>B</sub> check finished

## 3.3 Revision History

Table 63 Revision History

Reference	Change to Previous Version	Change Request Comment
<b>V1.1.0.1.14</b>		
<b>Chapter 3.1.1.7.7</b>	Text modified to better clarify DMU_HF_PROCONDBG.DBGIFLCK bit and debug password handling (documentation fix only, no change in implementation)	0000049759-179
<b>Chapter 3.1.3.2.5</b>	CAN BSL Init message 2 - DBPM=9 is an invalid value	0000058016-29
<b>Chapter 3.1.1.6</b>	Note added to the description of Boot Mode Header evaluation sequence to better clarify UCB_BMHD_ORIGINAL/COPY handling (documentation fix only, implementation is correct)	0000052718-169

## AURIX™ TC3xx Platform Firmware

**Table 63 Revision History**

<b>Reference</b>	<b>Change to Previous Version</b>	<b>Change Request Comment</b>
<b>Chapter 3.2.1.2</b>	Note added explaining that EMEM-related checks are performed by CHSW according to EMEM module presence on the product (assured by CFS content of the CHSW Reference Tables)	0000057932-17
<b>Table 53</b>	Correction in Table showing checks/bits assignments (documentation fix only, no change in implementation)	0000058016-26
<b>V1.1.0.1.15</b>		
	No change	
<b>V1.1.0.1.16</b>		
<b>Chapter 3.2.1.1</b>	Typo corrected in footnote 4) to SCU_STMEM1 register description - now stating "SWAP_CFG" instead of as wrongly before "SWAP_INDEX" (documentation fix only, no change in implementation)	0000058016-72
<b>Chapter 3.2.1.1</b>	Description of the reserved bits in SCU_STMEM1 register modified to avoid misunderstanding, that they must be always zero after start-up (documentation fix only, no change in implementation)	0000058016-73
<b>V1.1.0.1.17</b>		
<b>Chapter 3.1.2.2</b>	Note added, explaining FW handling after LBIST execution (documentation improvement only, no change in implementation)	0000058016-76

## On-Chip System Connectivity {and Bridges}

### 4 On-Chip System Connectivity {and Bridges}

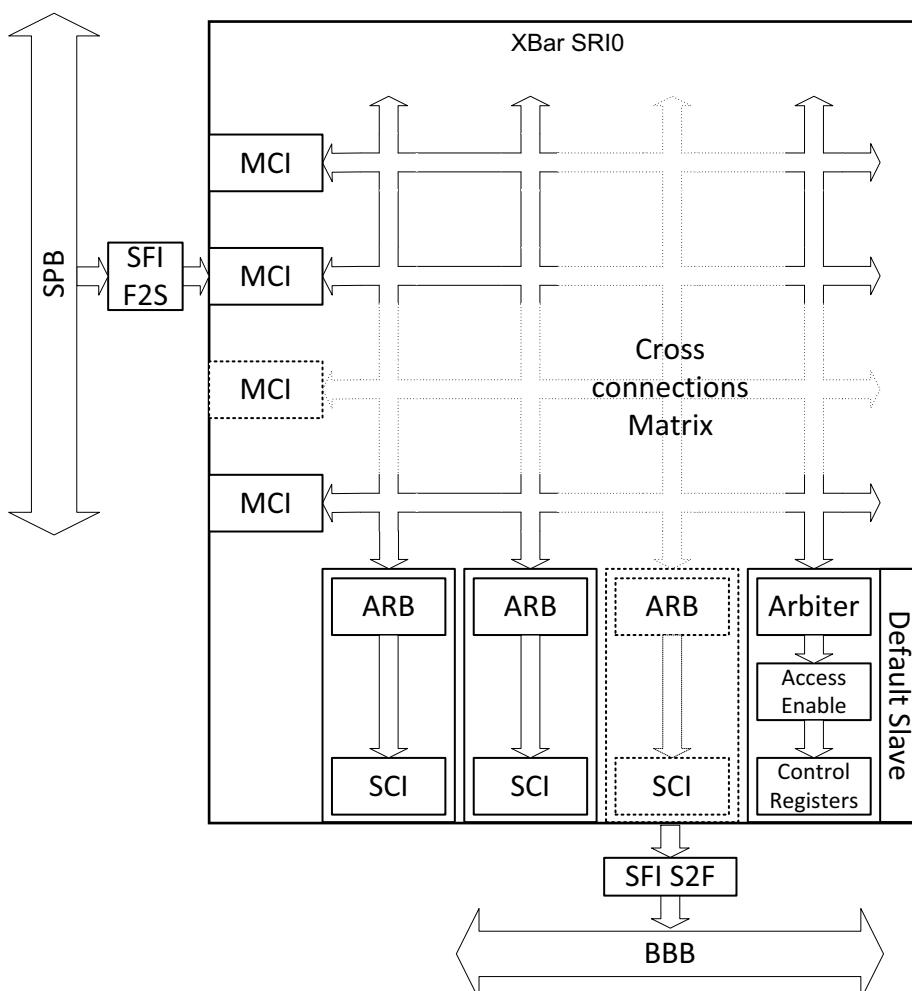
The AURIX™ TC3xx Platform has three independent on-chip connectivity resources:

- System Resource Interconnect Fabric (SRI Fabric)
- System Peripheral Bus (SPB)
- Back Bone Bus (BBB)

The SRI Fabric connects the TriCore CPUs, the DMA module, and other high bandwidth requestors to high bandwidth memories and other resources for instruction fetches and data accesses.

The SPB connects the TriCore CPUs, the DMA module, and other SPB masters to the medium and low bandwidth peripherals. SPB masters do not directly connect to the SRI Fabric, and will access SRI attached resources via a SFI\_F2S Bridge: see [Section 4.6](#).

The BBB connects the TriCore CPUs, the DMA module, and SPB masters with ADAS resources. SRI Masters do not directly connect to the BBB, but access BBB attached resources via a SFI\_S2F Bridge: see [Section 4.7](#). SPB masters also do not directly connect to the BBB, but access BBB attached resources via bridging over the SRI Fabric.



**Figure 25 Block diagram**

## On-Chip System Connectivity {and Bridges}

### 4.1 Feature List

The System Resource Interconnect (SRI) is the connectivity fabric for TriCore CPUs and high performance modules such as the DMA. A key component of the fabric is the SRI crossbar, which connects all the agents in one SRI domain. The SRI crossbar carries the transactions between the SRI Masters and SRI Slaves of the domain.

The SRI crossbar supports parallel transactions between different SRI Master and SRI Slave agents. In addition to the parallelism of concurrent requests, it also supports pipelined requests from an SRI Master to a SRI Slave.

#### SRI features

SRI feature overview:

- Single and burst read and write transactions (up to 4 x 64bit bursts)
- Atomic Read Modify Write transactions
- Pipelined transactions from SRI Masters to SRI Slaves
- Arbiter for each SRI Slave, with individual configuration
  - two round-robin groups, high and low priority
  - control of bandwidth to the high priority group
- EDC (Error Detection Code) on all address and control information transferred from SRI Master to SRI Slave
  - EDC on all data and control information transferred from SRI Master to SRI Slave for writes and RMWs
  - EDC on all data and control information transferred from SRI Slave to SRI Master for reads and RMWs

#### 4.1.1 What is new in the SRI Fabric

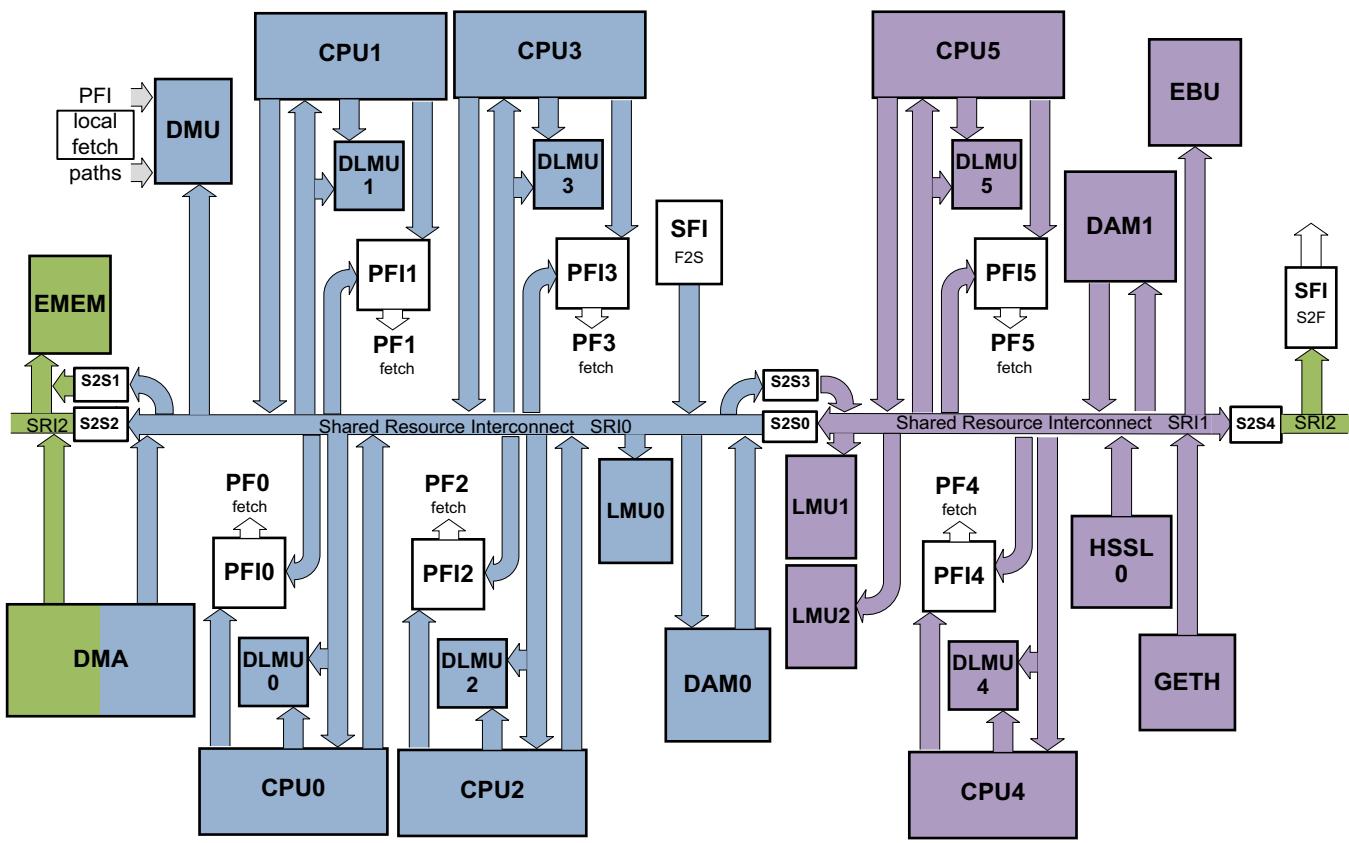
Major differences of the AURIX™ connectivity compared to previous AURIX™ based products.

- SRI Fabric can now consist of one or multiple independent crossbars
- SRI Crossbar arbitration scheme simplified to two layer round-robin

### 4.2 Overview

The SRI Fabric consists of one or more crossbars which support single and burst data transfers. If there are multiple crossbars, they are connected by bridges (S2S bridges: see [Section 4.5](#)).

## On-Chip System Connectivity {and Bridges}

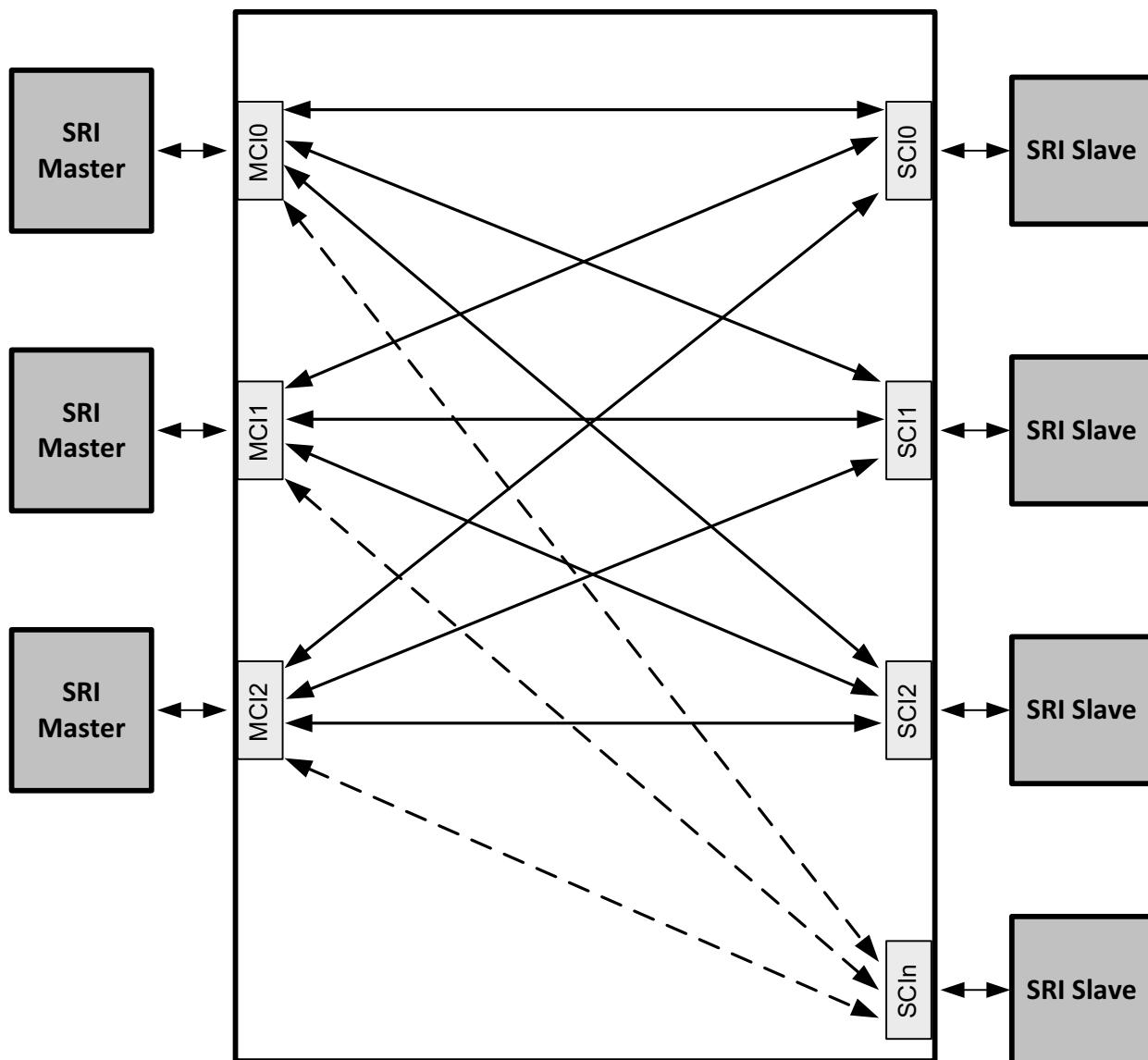


**Figure 26 Overview of the different SRI domains**

SRI agents (masters and slaves) which are connected to the same crossbar form an SRI domain. There will be at least one and possibly more SRI domains in AURIX™ TC3xx Platform family members. As an example the TC39xB has three domains: Domain 0 with 4 CPUs, Domain 1 with 2 CPUs, and Domain 2 with ADAS and debug functionality. S2S bridges are special in that they are present in two domains.

Due to the S2S bridging, all SRI Masters can directly address (access) most SRI Slaves, regardless of the SRI domain to which the master or slave is directly attached. Note: the reason for stating most slaves, rather than all slaves, is that even within the a single SRI domain, there may be masters and slaves which are not connected, since there is no functional need to communicate.

## On-Chip System Connectivity {and Bridges}



**Figure 27 SRI crossbar point to point connection scheme**

The SRI crossbar provides SRI Master Connection Interfaces (MCI<sub>x</sub>) to connect SRI Master modules and SRI Slave Connection Interfaces (SCI<sub>x</sub>) to connect SRI Slave modules to the SRI crossbar. There is one arbiter per connected SRI Slave module and the infrastructure for the enabled read/write data paths.

The SRI Fabric will always include at least one slave that provides housekeeping functions. The major purpose of these functions is to allow access to the SRI Fabric control and status registers. The second purpose is to respond (with an error) to all SRI transactions which address no other SRI Slave.

Please note that only those SRI Master to SRI Slave connections are implemented that are required for the system functionality; for example, an S2S master from another domain does not need to connect to an S2S slave connecting back to the same domain.

The SRI Fabric provides arbitration that allows the configuration of SRI Master priorities to be different for every SRI Slave. For support of system level diagnosis the SRI Fabric includes resources to capture SRI Error and SRI Transaction ID errors.

## On-Chip System Connectivity {and Bridges}

**Table 64 SRI Fabric Terms**

Term	Description
Agent	An SRI agent is any master or slave device which is connected to the SRI Fabric.
Master	An SRI master device is an SRI agent which is able to initiate transactions on the SRI Fabric.
Slave	An SRI slave device is an SRI agent which is not able to initiate transactions on the SRI. It is only able to respond to transactions operations that are directed to it by the SRI Fabric.
SRI crossbar	The SRI crossbar provides the interconnects between Masters and Slaves in the same domain. The SRI crossbar includes arbitration mechanisms and error capture capabilities.
MCI	Each Master is connected via one Master Connection Interface. The SRI Fabric contains control and status registers which affect MCI priority and provide related error information.
SCI	Each Slave is connected via one Slave Connection Interface. The SRI Fabric contains control and status registers which include control and error informations related to the SCI.
Domain	An SRI domain consists of those agents which are connected to a specific SRI crossbar. There will be at least one Master (or S2S bridge acting as a Master) and at least one Slave (or S2S bridge acting as a Slave), and an instance of a crossbar providing full or partial connectivity between all the agents in the domain.
Arbiter	If two (or more) Masters attempt to access the same Slave, the arbiter provides the decisions as to the order in which Masters gain access. The order is determined by the two-level round-robin mechanism implemented in the arbiter and the configuration programmed by the user.

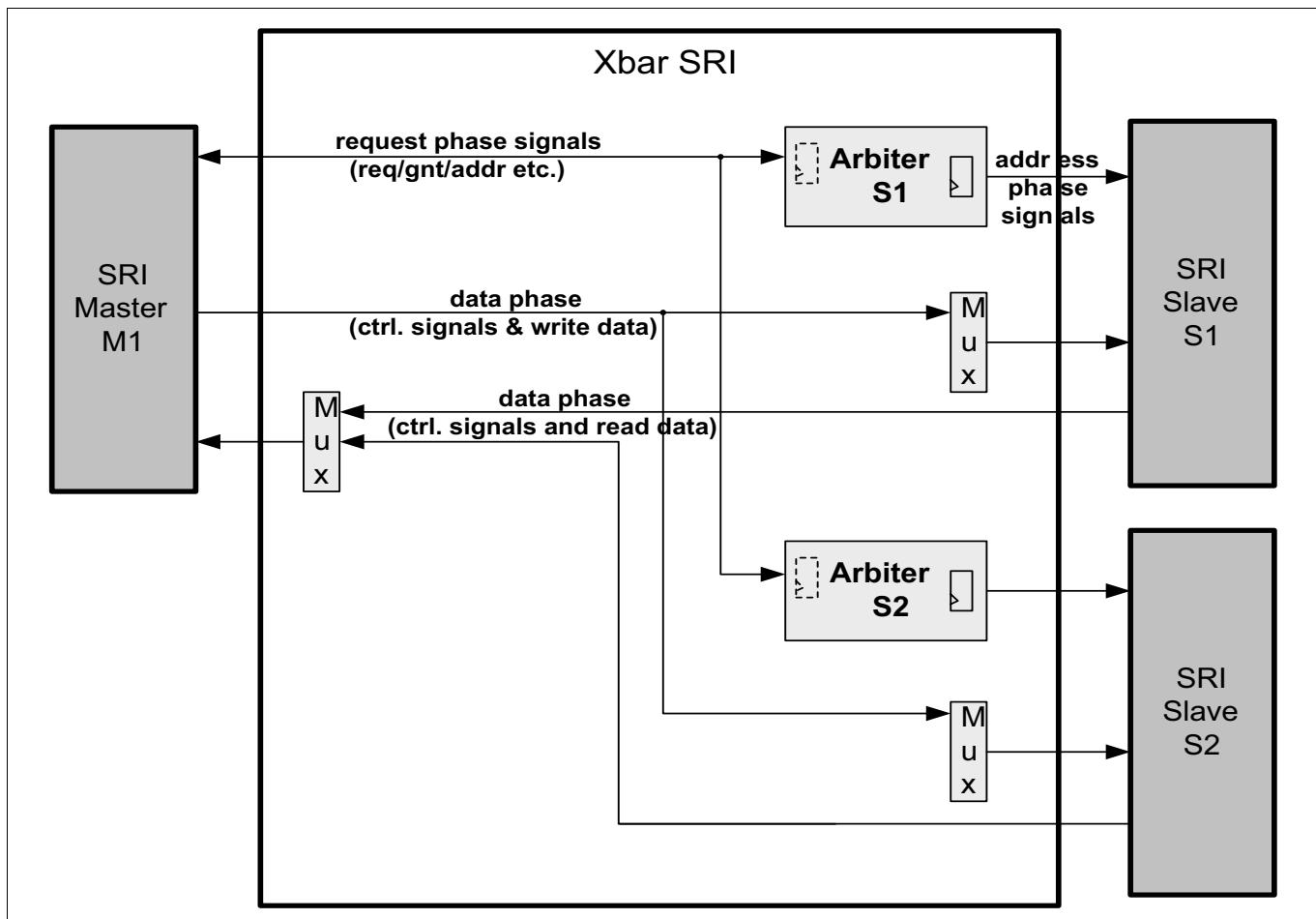
### 4.3 Functional Description

For configuration and debug, a certain amount of detail about the working of the SRI Fabric is required. Following is sufficient functional information to perform those tasks.

#### 4.3.1 Operational Overview

This section describes the functionality of the SRI crossbar module in order to enable the user to configure the SRI Fabric control registers, and to interpret the status and error capture resources.

## On-Chip System Connectivity {and Bridges}



**Figure 28 Example SRI crossbar connections between one SRI Master and two SRI Slaves**

The SRI crossbar can perform several transactions for different masters in parallel if the masters are accessing different slaves.

### 4.3.1.1 Master Connection Interface (MCI)

Each SRI Master in the system is mapped to master related control and status register bits and bit fields via its MCI number (see [Section 4.3.4](#)).

The MCI number is completely separate from the Master Tag(s) generated from the SRI Master connected to that MCI. However, the set of Master Tags presented by a specific MCI is fixed for a given device.

### 4.3.1.2 Slave Connection Interface (SCI)

Each SRI Slave in the system is mapped to slave related control and status register bits and bit fields via its SCI number (see [Section 4.3.4](#)).

### 4.3.1.3 Slave Arbiter

Each SCI has an associated arbiter which includes all the functionality for the following tasks:

- Arbitration: selection of the winning master when multiple requests are concurrent.
- Error capture: errors detected by a Slave (some errors are detected by Masters) will have transaction information captured where possible and the condition (if enabled) signalled to the system via the interrupt router (INT).

## On-Chip System Connectivity {and Bridges}

### 4.3.1.4 Default Slave

A default slave is an SRI Slave which implements several necessary housekeeping functions. As an SRI Slave, it has its own SCI and arbiter. A default slave may service a single or multiple domains. In a multiple domain system there will be more than one default slave. The allocation of default slaves to domains is fixed and its SCI number cannot be changed. A default slave provides these functions:

- access to some or all of the SRI Fabric control and status registers.
- all transactions that are accesses to non-existent address regions<sup>1)</sup> in a domain (if there are multiple domains), or the complete system (if there is a single domain), are provided to a default slave for a clean error response.
- implementation of the OLDA region (see next section).

For accesses to the control and status registers, only single data transactions of word size (32-bit) are supported. Any other transaction types are not supported. Unsupported transactions are acknowledged with an error. Each default slave will have its own valid address space, which is used for accesses to its allocated Fabric control and status registers. For a detailed description of the registers see [Chapter 4.4](#).

A default slave finishes SRI transactions to non-existent addresses (both reads and write) with an error, which will activate the error capturing mechanism of the default slave's arbiter.

**Note:** *A read from an SRI Master to a non-existent address on the BBB may be handled both by the Bus Control Unit on the BBB, and also reported on the SRI Fabric by the SFI\_S2F\_Bridge. In the SRI Fabric, this transaction information will be captured by the SFI\_S2F\_Bridge's arbiter, rather than the default slave for the domain. A write from an SRI Master to a non-existent address on the BBB will be posted and only directly reported on the BBB, and may be indirectly reported to the system otherwise.*

### 4.3.1.5 OnLine Data Acquisition (OLDA)

A default slave may also provide OnLine Data Acquisition (OLDA) region support. The OLDA is an address space where writes can complete without error but no memory is addressed. This allows production code to be written which writes data to memory, where the memory is only present in the Emulation Device. When running on an Emulation Device, the user can map EMEM to the OLDA region address space using the memory overlay feature, and the write data is then stored in the EMEM. When running in the Production Device, a default slave (when OLDA is enabled) terminates writes to the OLDA address space without error, even though no memory exists at the target address, and the write data is discarded.

If OLDA support is enabled in a default slave, direct write accesses (without redirection) to the OLDA range are not really executed, and they do not generate a bus error trap<sup>2)</sup>. If OLDA support is not enabled, write accesses will generate a bus error trap.

OLDA support is enabled by setting bit OLDAEN to 1 in a domain 0 BRCON register.

The base address of the virtual OLDA memory range(s) is defined in the memory map chapter. Accesses to the OLDA range are also supported in cached address space for all masters, but consider the footnote for the consequences for CPUs on cached write accesses<sup>2)</sup>.

Read accesses to the OLDA range generate a bus error trap, if not redirected to a physically available overlay block. Successful accesses to the OLDA memory range will only take place when the accesses are redirected to real, physical memory<sup>3)</sup>.

1) non-existent address regions = all Reserved address ranges described in the Memory Map chapter.

2) Write accesses to a cached memory address will trigger a read to fill the cache line before the data is written to the cache. This read will trigger a bus error.

3) This is mentioned for completeness as it is not really intended usage.

## On-Chip System Connectivity {and Bridges}

Note that switching on or off the OLDA function takes a finite amount of time after the register write completes. It is therefore possible for an immediately subsequent access to be pipelined to the default slave providing the OLDA region, and be acknowledged with either the “on” or “off” behavior. Accesses from the same master can be simply synchronised by a read back of the modified control registers before any accesses of interest; however for other masters more specific synchronisation is required.

**Note:** *In OTARx registers, any target address can be selected for redirection, including addresses in the OLDA range. However, the handling of direct accesses to the OLDA range is completely controlled in a default slave.*

### 4.3.1.6 Control and Status Register Access Protection

The SRI Fabric control and status registers are protected by a master TAG ID-based access protection mechanism. Each on-chip resource with direct or indirect bus master capability has a unique master TAG ID that can be used to identify the master of an on-chip bus transaction.

Access Enable:

TAG ID based protection means that on-chip write access to the SRI control and status registers can be disabled for each master TAG ID individually (with the exception of the Access Enable registers themselves, which are Safety Endinit protected). For a disabled master TAG ID, access will be terminated with an error acknowledge.

Access Enable Registers (ACCEN1/0) have these properties:

- define which master TAG ID is allowed write access to SRI control and status registers
- are Safe Endinit write protected

After reset, all ACCEN1/0 access enable bits and access control bits are enabled. The access protection mechanism must be configured to put the system into what is required for a safe operation.

### 4.3.2 Arbitration Details

Each SCI arbiter has access to all arbitration/address phase signals from the MCIs it is connected to, and views all requests from MCIs in parallel.

#### 4.3.2.1 Master Request Arbitration

The underlying arbitration mechanism is round-robin, which is simple to understand and is starvation-free. The general concept of round-robin is granting requests to masters in a circular order. The implemented round-robin mechanism is sometimes referred to as ‘round-robin work conserving’ as requests are not delayed if the ‘nominal winner’ is inactive; see **Round-robin Groups** below for more detail.

To support real-time behavior with more control over latencies of some Masters’ transactions, this scheme is enhanced. The specific arbitration supports two round-robin priority levels - a high priority level and a low priority level. Each level implements a round-robin arbitration scheme between participants in that level. In addition, the number of transactions continuously granted to the high priority level can be increased by use of the High Priority Round Share parameter; see **Two-Round Interaction** below for details.

#### Arbitration Configuration

The arbiter for each SCI is configurable for the priority of those MCI to which it is connected: see **Chapter 4.4.2**. The priorities (low or high) of each connected MCI can be programmed for each arbiter individually, via the arbiter PRIORITY register: see **Register "PRIORITY $x$  (x=0-15)"**. The priority of each master is defined by a single bit in the register that corresponds to its MCI number in the domain. This per-arbiter configuration allows a given master to be handled with different priorities for accesses to different slaves.

## On-Chip System Connectivity {and Bridges}

After reset most enabled MCIs are set to the low priority level, although bridges and DMAs are typically set to high priority.

For error diagnosis purposes, the arbiter samples all the necessary transaction information and provides this in [Register "ERRADDRx \(x=0-15\)"](#) and [Register "ERRRx \(x=0-15\)"](#) if an SRI protocol error occurs.

### Round-robin Groups

If more than one master is mapped to the priority of a round-robin group, the requests of the mapped masters will be handled by the round-robin algorithm.

After the winner of a round-robin group is granted a transaction, the round-robin group starts a new arbitration round. In this round the next highest MCI ID number which is requesting a transaction will win the arbitration. If there is no requesting MCI with a higher ID number in the round-robin group, the algorithm will consider the lowest MCI ID number in the group that is requesting, followed by the next higher MCI ID number and so on.

The mechanism works by selecting each requesting master slot in turn per arbitration cycle, in ascending order: MCIO to MCIx-1 (for x MCIs in the domain). If a master doesn't have a request in the current cycle of arbitration then it is bypassed for the next slot, and so on. Thus, if only one master is requesting over a given time period, that master will be selected consecutively during multiple arbitration rounds, without incurring any latency.

### Two-Round Interaction

In arbitrating between high priority masters, the low priority MCIs are collectively viewed as a request. After at most 'High Priority Round Share' consecutive high priority master transactions, the low priority round is granted an access. The evaluated winner of the low priority round then carries out its transaction. If there are no high-priority MCIs, or no low-priority MCIs, the scheme operates as a single-level round-robin arbitration.

### Request Latency

If no other transaction is pending, a transaction from an MCI commences in the next clock. If there are multiple requests pending, or arrive at the same time, then the latency depends on whether the MCI is in the high or low priority round share, and how many other requestors are in each round. [Table 65](#) shows the worst case number of transactions that a Master might have to wait before its transaction can commence in a domain with the maximum of 8 Masters.

**Table 65 Master Request Latency**

Number of High Priority/ Low Priority Masters	High Priority Round Share	Worst case delay in arbitrations for a High Priority Master	Worst case delay in arbitrations for a Low Priority Master
0 / 8	(0)	N.A.	7
1 / 7	1	1	13
	2	1	20
	3	1	27
	4	1	34
	5	1	41
	6	1	48
	7	1	55

## On-Chip System Connectivity {and Bridges}

**Table 65 Master Request Latency**

Number of High Priority/ Low Priority Masters	High Priority Round Share	Worst case delay in arbitrations for a High Priority Master	Worst case delay in arbitrations for a Low Priority Master
2 / 6	2	2	17
	3	2	23
	4	2	29
	5	2	35
	6	2	41
	7	2	47
3 / 5	3	3	19
	4	3	24
	5	3	29
	6	3	34
	7	3	39
4 / 4	4	4	19
	5	4	23
	6	4	27
	7	4	31
5 / 3	5	5	17
	6	5	20
	7	5	23
6 / 2	6	6	13
	7	6	15
7 / 1	7	7	7
8 / 0	(0)	7	N.A.

The latency for domains with fewer masters can be computed simply:

If there is only a single round of arbitration (high or low), then

```
max_delay = number_of_masters - 1
```

If there are two rounds (high and low) of arbitration, then for a high priority master:

```
HP_max_delay = number_of_high_priority_masters
```

and for a low priority master:

```
LP_max_delay = ((HPRS+1) x number_of_low_priority_masters) - 1
```

The worst case number of clock cycles a Master might have to wait will depend on the response time of the respective slave. So, if for example, the SRI Slave has a response time of 10 clocks, the worst case wait in clock cycles for a low priority master in a 2 high, 6 low configuration with a high priority share set to 7 is 470 clocks. Usually SRI slaves will have a deterministic response time to transactions (although it might vary between read and write transactions, and the data width to be transferred). It is typically only in the case of off-chip accesses (such as the EBU) where there might be high variation in transaction times.

## On-Chip System Connectivity {and Bridges}

### 4.3.3 SRI Errors

SRI Error conditions can be detected by both masters and slaves depending upon the condition. Errors are reported via alarms to the SMU or interrupts or traps taken by a CPU.

For some errors, a slave can signal the corresponding master that an error has happened, which results in an immediate termination of the current transaction. For other errors during a transaction, the transaction continues, but the error is tracked by the SRI Fabric, and may be signalled via an alarm or an interrupt if enabled in both the SRI Fabric and interrupt router (INT).

There are three error types:

- SRI Protocol Errors
- SRI Transaction ID Errors
- SRI EDC Errors

Note: the term EDC error refers here to the combined set of SRI EDC features (SRI Address phase EDC, SRI Write Data EDC and SRI Read Data EDC). Although the encoding scheme could allow for correction of single bit errors, the decoders are configured to only use detection as this provides a higher coverage against multi-bit errors.

#### 4.3.3.1 SRI Protocol Errors

The following error conditions are detected by SRI Slaves and result in error termination of the transaction (SRI protocol error) to the SRI Master and/or an alarm:

- non-existent address region accessed by an SRI Master<sup>1)</sup>
- reserved address within an existing memory region
- unsupported or reserved opcodes
- access level is incorrect (supervisor register accessed in user mode)
- access protection is configured to not allow the current SRI Master read access<sup>2)</sup>
- access protection is configured to not allow the current SRI Master write access<sup>3)</sup><sup>4)</sup>
- unsupported operation to a specific register
- EDC invalid for the address phase

For testing purposes, an SRI Protocol Error can be generated by the application software with an access to a reserved address (see chapter Memory Map).

Those SRI Slaves which have detected a protocol error can be determined by inspection of [Register "PESTAT"](#).

#### 4.3.3.2 SRI Transaction ID Errors

A transaction ID is an identifier connected to all phases of a transaction, in order to make the transaction unique in the SRI Fabric during the transaction's lifetime.

The transaction ID is used by SRI Masters and SRI Slaves to identify errors (usually transient) in the SRI Fabric that might result in data packets received by a master or slave that do not correspond to the started transaction. If the read/write transaction identifier doesn't match the transaction identifier sent during the address phase, this is a Transaction ID error and tracked by the SRI Fabric.

In addition to SRI Fabric errors, forcing a Transaction ID Error is used in situations where at the data source side (the SRI Master for write transactions and the SRI Slave for read transactions), a data phase has to be invalidated

1) The accesses to non-existent address regions are checked by a default slave.

2) Read protection is intended to protect memories and not SFR ranges.

3) MiniMCDS SFR range is not protected. Debug functionality.

4) Some slaves don't issue a bus error. In that case the write is prevented and an alarm is issued. This is the case for CPU, LMU, EBU and EMEM

## On-Chip System Connectivity {and Bridges}

(e.g. detection of an uncorrectable error from an accessed memory). In these situations, an invalid transaction ID is sent in order to invalidate the data phase.

For testing purposes, an SRI Transaction ID error condition can be generated by injecting a non-correctable error into one of the SRAMs (e.g. CPU Scratch Pad SRAM), and then reading the corrupted data by an SRI Master.

Those SRI Masters and SRI Slaves which have detected a Transaction ID error can be determined by inspection of **Register "TIDSTAT"**.

### 4.3.3.3 SRI EDC Errors

The SRI Fabric provides EDC protection for the:

- Address phase of an SRI transaction
- Transmitted read and write data phases

#### SRI Address Phase EDC Errors

- If an SRI Slave detects an SRI address phase EDC error, it finishes the transaction with SRI error acknowledge (a protocol error) and does not process it further
- In addition to the protocol error, each SRI Slave signals an address EDC error to the SMU as an alarm

See the **Error Handling** section for details on Protocol Error configuration and information capture.

#### SRI Write Data EDC Errors

An SRI Slave will check each data phase of a write transaction, (1, 2 or 4 phases depending on the transaction length) for EDC validity.

If an SRI Slave detects an SRI write data EDC error, it is signalled to the SMU as an alarm.

Whether a SRI Slave merely monitors the EDC error condition, or attempts to prevent the state update, is not defined by the SRI infrastructure, but should be described by each SRI Slave Agent.

#### SRI Read Data EDC Errors

An SRI Master will check each data phase of a read transaction, (1, 2 or 4 phases depending on the transaction length) for EDC validity.

If an SRI Master detects an SRI read data EDC error, it is signalled to the SMU as an alarm.

Whether an SRI Master merely monitors the EDC error condition, or attempts to prevent the usage of the returned data is not defined by the SRI infrastructure, but should be described by each SRI Slave-Agent.

Because of their key role, the behavior of CPU is reiterated here. A trap will be taken by the CPU if a received 64-bit phase of data is used by the CPU. A phase might not be used if a cache line is requested and the beat containing the EDC error is not required for the instruction stream or the data load instruction. If any part of a received memory line contains an EDC error, a possible cache update is aborted and the received data is discarded.

### 4.3.3.4 Error Handling

If an arbiter recognises that its associated SRI Slave has signalled an SRI Protocol error for a transaction, the arbiter samples all relevant information of the transaction in the diagnostic registers (**ERRx (x=0-15)** and **ERRADDRx (x=0-15)**). An indication of this error can be provided to the interrupt router (INT), and this indication can be controlled, via the **PECONx (x=0-15)**.PEEN control bit. Only when this bit is set is the error information captured and the trigger condition signalled to the interrupt router.

## On-Chip System Connectivity {and Bridges}

### Notes

1. The two error registers **ERRx (x=0-15)** and **ERRADDRx (x=0-15)** in each arbiter are updated with the content of the currently processed transaction in the data phase. The registers are only updated if they are not locked due to an earlier protocol error.

The stored information can be read from the arbiter via word reads.

The error capture registers in the arbiter will not be updated again until the indication is acknowledged to the slave arbiter module by software (set the respective **PECONx (x=0-15).PEACK**).

The error capture registers will be locked after the first error. This results in the non capture of subsequent error information from the same SRI Domain, until the write releasing the lock is processed.

A master or a slave can detect a transaction ID error. An indication of this error can be provided to the interrupt router (INT), and this indication can be controlled via the **TIDEN.ENMCI** and **TIDEN.ENSCI** control bits. Only if the relevant enable bit is set is the error condition captured in **TIDSTAT**, and the trigger condition signalled to the interrupt router.

### Notes

1. While a status bit for a error source is set in either the **PESTAT** or **TIDSTAT** register, a new error from this particular source doesn't generate a new indication.
2. Each agent in the SRI Bus system has two error conditions: protocol errors, and Transaction ID errors. All errors from an SRI Bus domain share a single SRN in the interrupt router (INT) for each SRI Domain.

### 4.3.3.5 Error Tracking Capability

The SRI crossbar tracks all SRI transactions for SRI protocol errors. In addition it tracks information about SRI transaction ID errors. This is done by all arbiters in parallel, since the SRI crossbar supports the processing of multiple transactions from masters and slaves in parallel, which can result in concurrent events at different SCIs.

For this purpose, each arbiter has two registers where it samples the transaction information of the transaction where the first protocol error happened.

**Note:** *Protocol errors will lock the error registers. All subsequent error events will not be captured in the error registers until the lock is released.*

Further protocol errors will be ignored by an arbiter that has detected a protocol error until the capture mechanism is re-activated via the arbiter internal control register. A detected protocol error is signalled by the arbiter to the per-domain error status register **PESTAT**. Each error signal can be masked individually by the **PECONx (x=0-15)** control registers in the arbiter modules for SCIx. In addition the **ERRx (x=0-15)** and **ERRADDRx (x=0-15)** registers are accessible in each SCIx.

For transaction ID errors of write transactions, the SCIs propagate errors to the associated bit in the lower half of the per-domain register **TIDSTAT**. For transaction ID errors of read transactions, the MCIs propagate errors to the assigned bit in the upper half of the same register.

Once an error is signalled from an arbiter or an MCI/SCI to the system, the system can read out the error status registers to find out which arbiter(s) or master/slave have detected an error. The system can then perform more detailed diagnostics by reading out the error registers in the respective arbiter for a protocol error. The error information captured for an SRI protocol error allows the identification of the master via the sampled master tag ID, and the final destination via the captured target address. In addition, the **ERRx (x=0-15)** register samples the opcode and the read, write, and supervisor signals of the transaction.

In addition to the pure transaction information, the SRI **ERRx (x=0-15)** captures some additional information. In the AURIX™ family, this additional information is only provided by the DMA peripheral, to indicate the channel performing the transaction. **Table 66** shows the encoding of the MCI\_SBS bit field for the AURIX™ family.

## On-Chip System Connectivity {and Bridges}

**Table 66 Encoding of MCI\_SBS in ERRx (x=0-15)**

MCI_SBS[7:0]	Bit field encoding
MCI_SBS[7:0]	This field is only defined if the master TAG ID in the TR_ID field is a DMA hardware resource group. MCI_SBS[7] provides the DMA Move Engine number MCI_SBS[6:0] provides the DMA channel number

After reading all relevant error information, the error capture mechanism should be re-enabled.

### 4.3.3.6 Indication Event Interactions

There are some interactions between indication events. In general, several indications from the same, or a different source (arbiter, MCI or SCI) can occur at the same time. One indication could occur several times before a service request routine services it.

The following examples describe the system behavior without rearming from a service routine when all consecutive indications/events come from the same domain.

#### Two Consecutive Protocol Errors

The first protocol error is captured, together with the generation of an indication to the system. The second protocol error is not captured, since the registers ERR and ERRADD are already locked, and no indication is generated.

#### Two Consecutive Transaction ID Errors

The first transaction ID error generates an indication to the system. The second transaction ID error will not generate an indication to the system.

### 4.3.3.7 Releasing the lock from registers ERR and ERRADD

If the ERR and ERRADD registers are locked. When the registers are locked, the lock can be released by writing a 1 to PECONx.PEACK.

### 4.3.4 Implementation of the SRI Fabric

This chapter describes the SRI Fabric implementation in the AURIX™. The knowledge of the specific implementation (e.g. the connection of the SRI Master / SRI Slave devices to the Fabric) is necessary in order to:

- define the arbitration priority for masters (using their MCI number)
- map error information to the connected slave devices (using their SCI number)
- map SRI crossbar (arbiter) control registers to connected slave devices

The first part of the chapter includes tables for each SRI domain that describes:

- the mapping of SRI Master devices to MCIs
- the mapping of SRI Slave devices to SCIs
- the implementation of specific connections between MCIs and SCIs

#### Notes

1. *Each CPUx (x = 0...5) has only a single SRI Master connection, where both code fetches and data accesses are presented (labelled CPUx in MCI tables).*

## On-Chip System Connectivity {and Bridges}

2. Each CPU $x$  ( $x = 0 \dots 5$ ) has both an SRI Slave connection for access to its associated PFlash (labelled CPU $xP$  in SCI tables), and an SRI Slave connection for access to its associated SRAMs, SFRs and CSFRs (labelled CPU $xS$  in SCI tables).

### 4.3.4.1 Mapping of SRI Masters to Domain 0 Master Interfaces

Derivative	MCI0	MCI1	MCI2	MCI3	MCI4	MCI5	MCI6	MCI7	MCI8	MCI9	MCI10	MCI11	MCI12
<b>TC39xA</b>	DMA MIFO	SFI F2S	CPU0	CPU1	CPU2	CPU3	S2SO D1D0	Not Present	Not Present	Not Present	Not Present	Not Present	Not Present
<b>TC39xB</b>	DMA MIFO	SFI F2S	CPU0	CPU1	CPU2	CPU3	S2SO D1D0	Not Present	HSSLO	Not Present	Not Present	Not Present	Not Present
<b>TC3Ex</b>	DMA MIFO	SFI F2S	CPU0	CPU1	CPU2	CPU3	Not Present	Not Present	HSSLO	GETH	Not Present	Not Present	Not Present
<b>TC38x</b>	DMA MIFO	SFI F2S	CPU0	CPU1	CPU2	CPU3	Not Present	Not Present	HSSLO	GETH	Not Present	Not Present	Not Present
<b>TC37xEXT</b>	DMA MIFO	SFI F2S	CPU0	CPU1	CPU2	Not Present	Not Present	Not Present	HSSLO	GETH	Not Present	Not Present	GETH1
<b>TC37x</b>	DMA MIFO	SFI F2S	CPU0	CPU1	CPU2	Not Present	Not Present	Not Present	HSSLO	GETH	Not Present	Not Present	Not Present
<b>TC36x</b>	DMA MIFO	SFI F2S	CPU0	CPU1	Not Present	Not Present	Not Present	Not Present	HSSLO	GETH	Not Present	Not Present	Not Present
<b>TC35x</b>	DMA MIFO	SFI F2S	CPU0	CPU1	CPU2	Not Present	Not Present	Not Present	Not Present	GETH	DMA MIF1	DMA MIF2	Not Present
<b>TC3Ax</b>	DMA MIFO	SFI F2S	CPU0	CPU1	CPU2	CPU3	Not Present	Not Present	HSSLO	GETH	HSSL1	Not Present	Not Present
<b>TC33xEXT</b>	DMA MIFO	SFI F2S	CPU0	CPU1	Not Present	Not Present	Not Present	Not Present	Not Present	GETH	DMA MIF1	DMA MIF2	Not Present
<b>TC33x</b>	DMA MIFO	SFI F2S	CPU0	Not Present									

**Figure 29 Mapping of AURIX™ SRI Masters to Domain 0 MCI**

## On-Chip System Connectivity {and Bridges}

### 4.3.4.2 Mapping of SRI Slaves to Domain 0 Slave Interfaces

Figure 30 shows the mapping of master devices of the AURIX™ family members to the Domain 0 SCIs.

Derivative	SCI0	SCI1	SCI2	SCI3	SCI4	SCI5	SCI6	SCI7	SCI8	SCI9	SCI10	SCI11	SCI12	SCI13	SCI14	SCI15
TC39xA	S2S3 DOD1	DMU	DAM0	CPU0P	CPU0S	CPU1P	CPU1S	CPU2P	CPU2S	CPU3P	CPU3S	LMU0	S2S2 DOD2	S2S1 DOD2	Not Present	Default Slave
TC39xB	S2S3 DOD1	DMU	DAM0	CPU0P	CPU0S	CPU1P	CPU1S	CPU2P	CPU2S	CPU3P	CPU3S	LMU0	S2S2 DOD2	S2S1 DOD2	Not Present	Default Slave
TC3Ex	Not Present	DMU	DAM0	CPU0P	CPU0S	CPU1P	CPU1S	CPU2P	CPU2S	CPU3P	CPU3S	LMU0	Not Present	Not Present	Mini MCDS	Default Slave
TC38x	Not Present	DMU	DAM0	CPU0P	CPU0S	CPU1P	CPU1S	CPU2P	CPU2S	CPU3P	CPU3S	LMU0	Not Present	Not Present	Mini MCDS	Default Slave
TC37xEXT	Not Present	DMU	DAM0	CPU0P	CPU0S	CPU1P	CPU1S	Not Present	CPU2S	Not Present	Not Present	Not Present	S2S0 DOD2	S2S1 DOD2	Not Present	Default Slave
TC37x	Not Present	DMU	DAM0	CPU0P	CPU0S	CPU1P	CPU1S	Not Present	CPU2S	Not Present	Not Present	Not Present	Not Present	Not Present	Mini MCDS	Default Slave
TC36x	Not Present	DMU	Not Present	CPU0P	CPU0S	CPU1P	CPU1S	Not Present	Not Present	Not Present	Default Slave					
TC35x	SFI_S2F BBB	DMU	Not Present	CPU0P	CPU0S	CPU1P	CPU1S	Not Present	CPU2S	EMEM 0	EMEM 1	LMU0	LMU1	Not Present	Not Present	Default Slave
TC3Ax	SFI_S2F BBB	DMU	Not Present	CPU0P	CPU0S	CPU1P	CPU1S	Not Present	CPU2S	Not Present	CPU3S	EMEM 4	EMEM 5	S2S0 DOD2	S2S1 DOD2	Default Slave
TC33xEXT	SFI_S2F BBB	DMU	Not Present	CPU0P	CPU0S	Not Present	CPU1S	Not Present	EMEM 0	Not Present	Not Present	Not Present	Not Present	Not Present	Not Present	Default Slave
TC33x	Not Present	DMU	Not Present	CPU0P	CPU0S	Not Present	Not Present	Not Present	Default Slave							

Figure 30 Mapping of AURIX™ SRI Slaves to Domain 0 SCI

## On-Chip System Connectivity {and Bridges}

### 4.3.4.3 Mapping of SRI Masters to Domain 1 Master Interfaces

[Figure 31](#) shows the mapping of master devices to the Domain 1 MCIs.

Derivative	MCI0	MCI1	MCI2	MCI3	MCI4	MCI5
TC39xA	HSSL0	S2S3 D0D1	CPU4	CPU5	Not Present	Not Present
TC39xB	GETH	S2S3 D0D1	CPU4	CPU5	Not Present	HSSL1

**Figure 31 Mapping of SRI Masters to Domain 1 Master Interfaces**

AURIX™ family members smaller than the TC38x do not have a Domain 1.

### 4.3.4.4 Mapping of SRI Slaves to Domain 1 Slave Interfaces

[Figure 32](#) shows the mapping of slave modules to the SRI crossbar Slave Interfaces (SCI0 - SCI15). Most of the SRI crossbar control registers are related to the SRI crossbar Slave Interfaces (SCI0 - SCI15) or the SRI crossbar Master Interfaces. Therefore it is important to know which AURIX™ TC3xx Platform SRI Slave device relates to which SRI crossbar Slave Interface or arbiter module.

Derivative	SCI0	SCI1	SCI2	SCI3	SCI4	SCI5	SCI6	SCI7	SCI8	SCI9	SCI10	SCI11
TC39xA	Default Slave	S2S0 D1D0	CPU4P	CPU4S	CPU5P	CPU5S	EBU	NO LMU1	NO LMU2	NO LMU3	DAM1	S2S4 D1D2
TC39xB	Default Slave	S2S0 D1D0	CPU4P	CPU4S	CPU5P	CPU5S	EBU	LMU1	LMU2	Not Present	DAM1	S2S4 D1D2

**Figure 32 Mapping of AURIX™ SRI Slaves to Domain 1 SCI**

AURIX™ family members smaller than the TC38x do not have a Domain 1.

### 4.3.4.5 Mapping of SRI Masters to Domain 2 Master Interfaces

[Figure 33](#) shows the mapping of master devices for the AURIX™ Domain 2 MCIs.

Derivative	MCI0	MCI1	MCI2	MCI3	MCI4
TC39xA	S2S2 D0D2	S2S1 D0D2	S2S4 D1D2	Not Present	Not Present
TC39xB	S2S2 D0D2	S2S1 D0D2	S2S4 D1D2	DMA MIF1	DMA MIF2
TC37xEXT	S2S0 D0D2	S2S1 D0D2	Not Present	DMA MIF1	DMA MIF2
TC3Ax	S2S0 D0D2	S2S1 D0D2	Not Present	DMA MIF1	DMA MIF2

**Figure 33 Mapping of AURIX™ SRI Masters to Domain 2 MCI**

## On-Chip System Connectivity {and Bridges}

### 4.3.4.6 Mapping of SRI Slaves to Domain 2 Slave Interfaces

Figure 34 shows the mapping of slave modules for the AURIX™ Domain 2 SCIs.

Derivative	SCI0	SCI1	SCI2	SCI3	SCI4	SCI5	SCI6
<b>TC39xA</b>	Default Slave	SFI_S2F	EMEM 0	EMEM 1	EMEM 2	EMEM 3	Mini MCDS
<b>TC39xB</b>	Default Slave	SFI_S2F BBB	EMEM 0	EMEM 1	EMEM 2	EMEM 3	Not Present
<b>TC37xEXT</b>	Default Slave	SFI_S2F BBB	EMEM 0	EMEM 1	EMEM 2	Not Present	Not Present
<b>TC3Ax</b>	Default Slave	Not Present	EMEM 0	EMEM 1	EMEM 2	EMEM 3	Not Present

Figure 34 Mapping of AURIX™ SRI Slaves to SCI

## 4.4 Registers

Figure 35 along with Table 67 show all the registers of the SRI Fabric. The absolute register address is calculated from the Module Base Address (see Appendix document) by adding the Offset Address of the specific register and (0x20\*i), where ‘i’ is the index of the associated SCI.

### SRI Fabric Register Overview

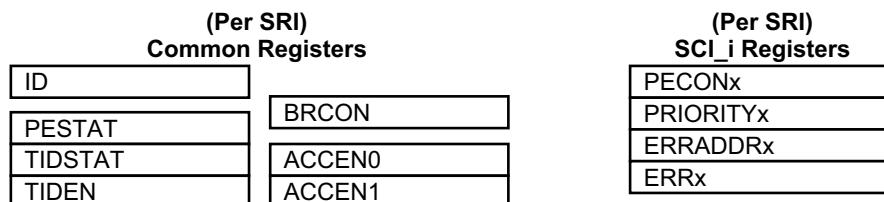


Figure 35 SRI Fabric Registers for all Domains

### Notes

1. Addresses listed in column “Offset Address” of Table 67 are word (32-bit) addresses on double word alignment. All odd-word addresses are reserved.
2. SRI control registers can be accessed only with SDTW (32-bit) transactions. 8-bit, 16-bit, 64-bit and RMW transactions are undefined.

Table 67 Register Overview - DOM (ascending Offset Address)

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
PECONx	Protocol Error Control Register x	00000 <sub>H</sub> + x*20 <sub>H</sub>	32,U,SV	32,P,SV	Application Reset	25
PRIORITYx	SCIx Arbiter Priority Register	00008 <sub>H</sub> + x*20 <sub>H</sub>	32,U,SV	32,P,SV	Application Reset	25
ERRADDRx	SCI x Error Address Capture Register	00010 <sub>H</sub> + x*20 <sub>H</sub>	32,U,SV	32,P,SV	Application Reset	26

## On-Chip System Connectivity {and Bridges}

Table 67 Register Overview - DOM (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
ERRx	SCI x Error Capture Register	00018 <sub>H</sub> + x*20 <sub>H</sub>	32,U,SV	32,P,SV	Application Reset	<a href="#">27</a>
ID	Identification Register	00408 <sub>H</sub>	32,U,SV	BE	Application Reset	<a href="#">20</a>
PESTAT	Protocol Error Status Register	00410 <sub>H</sub>	32,U,SV	32,P,SV	Application Reset	<a href="#">20</a>
TIDSTAT	Transaction ID Status Register	00418 <sub>H</sub>	32,U,SV	32,P,SV	Application Reset	<a href="#">21</a>
TIDEN	Transaction ID Enable Register	00420 <sub>H</sub>	32,U,SV	32,P,SV	Application Reset	<a href="#">22</a>
BRCON	Bridge Control Register	00430 <sub>H</sub>	32,U,SV	32,P,SV	Application Reset	<a href="#">22</a>
ACCENO	Access Enable Register 0	004F0 <sub>H</sub>	32,U,SV	32,SV,SE	Application Reset	<a href="#">23</a>
ACCEN1	Access Enable Register 1	004F8 <sub>H</sub>	32,U,SV	32,SV,SE	Application Reset	<a href="#">24</a>

## On-Chip System Connectivity {and Bridges}

### 4.4.1 Domain Common Registers

#### Identification Register

The identification register identifies the module, and provides revision and type information. The table below describes the identification register, which is implemented per domain.

ID															
Identification Register (00408 <sub>H</sub> ) Application Reset Value: 0004 D0XX <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUMBER															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	<b>Module Revision Number</b> This bit field defines the module revision number. The value of a module revision starts with 01H (first revision).
MOD_TYPE	15:8	r	<b>Module Type</b> The bit field is set to D0H which defines the module as a 32-bit module.
MOD_NUMBE R	31:16	r	<b>Module Number Value</b> This bit field defines a module identification number. The value for the SRI Fabric is 0004H.

#### Protocol Error Status Register

Note: Only the bits assigned to implemented SCIs are present. Bits corresponding to unimplemented SCIs are treated as reserved bits, which will be read as 0, and should be written with 0. See [Chapter 4.3.4](#) for which SCI is implemented.

#### PESTAT

PESTAT															
Protocol Error Status Register (00410 <sub>H</sub> ) Application Reset Value: 0000 0000 <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PESCI 15	PESCI 14	PESCI 13	PESCI 12	PESCI 11	PESCI 10	PESCI 9	PESCI 8	PESCI 7	PESCI 6	PESCI 5	PESCI 4	PESCI 3	PESCI 2	PESCI 1	PESCI 0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

## On-Chip System Connectivity {and Bridges}

Field	Bits	Type	Description
PESCIn (n=0-15)	n+16	rwh	<p><b>Protocol Error status of SCIn</b></p> <p>Writing 0 to the bit leaves the content unchanged, while writing 1 to the bit clears it. In case the bit is simultaneously cleared via software and set via hardware, the bit remains set and is not cleared.</p> <p>0<sub>B</sub> No protocol error has been indicated by SCIn 1<sub>B</sub> A protocol error has been indicated by SCIn</p>
0	15:0	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

### Transaction ID Status Register

#### Notes

- Only the bits assigned to implemented SCIs and MCIs are present. Bits corresponding to unimplemented SCIs and MCIs are treated as reserved bits, which will be read as 0, and should be written with 0. See [Chapter 4.3.4](#) for which MCIs and SCIs are implemented.
- Reset values for bits/bit fields referencing MCIs or SCIs that are not implemented or enabled are 0.

#### TIDSTAT

Transaction ID Status Register (00418 <sub>H</sub> ) Application Reset Value: 0000 0000 <sub>H</sub>																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
					TIDMC											
					I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
					r	rwh										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
TIDSCI	TIDSCI	TIDSCI	TIDSCI	TIDSCI	TIDSCI	TIDSCI	TIDSCI	TIDSCI	TIDSCI	TIDSCI	TIDSCI	TIDSCI	TIDSCI	TIDSCI	TIDSCI	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	

Field	Bits	Type	Description
TIDSCIIn (n=0-15)	n	rwh	<p><b>Transaction ID Error from SCIn Status</b></p> <p>Writing 0 to the bit leaves the content unchanged.</p> <p>Writing 1 to the bit clears it.</p> <p><i>Note:</i> If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.</p> <p>0<sub>B</sub> No transaction ID error has been indicated by SCIn 1<sub>B</sub> A transaction ID error has been indicated by SCIn</p>

## On-Chip System Connectivity {and Bridges}

Field	Bits	Type	Description
TIDMCIn (n=0-11)	n+16	rwh	<p><b>Transaction ID Error from MCIn Status</b>            Writing 0 to the bit leaves the content unchanged.            Writing 1 to the bit clears it.</p> <p><i>Note:</i> If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.</p> <p>0<sub>B</sub> No transaction ID error has been indicated by MCIn            1<sub>B</sub> A transaction ID error has been indicated by MCIn</p>
0	31:28	r	<p><b>Reserved</b>            Read as 0; should be written with 0.</p>

### Transaction ID Enable Register

#### Notes

- Only the bits assigned to implemented SCIs and MCIs are present. Bits corresponding to unimplemented SCIs and MCIs are treated as reserved bits, which will be read as '0', and should be written with '0'. See [Chapter 4.3.4](#) for implemented MCI and SCI per domain
- Reset values for bits/bit fields referencing MCIs or SCIs that are not implemented or enabled are zero.

#### TIDEN

#### Transaction ID Enable Register

(00420<sub>H</sub>)

Application Reset Value: 0FFF 3FFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				ENMCI 11	ENMCI 10	ENMCI 9	ENMCI 8	ENMCI 7	ENMCI 6	ENMCI 5	ENMCI 4	ENMCI 3	ENMCI 2	ENMCI 1	ENMCI 0
				r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENSCI 15	ENSCI 14	ENSCI 13	ENSCI 12	ENSCI 11	ENSCI 10	ENSCI 9	ENSCI 8	ENSCI 7	ENSCI 6	ENSCI 5	ENSCI 4	ENSCI 3	ENSCI 2	ENSCI 1	ENSCI 0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENSCIn (n=0-15)	n	rw	<p><b>Enable Transaction ID Error from SCIn</b>            0<sub>B</sub> Transaction ID errors from SCIn are not indicated            1<sub>B</sub> Transaction ID errors from SCIn are indicated</p>
ENMCIn (n=0-11)	n+16	rw	<p><b>Enable Transaction ID Error from MCIn</b>            0<sub>B</sub> Transaction ID errors from MCIn are not indicated            1<sub>B</sub> Transaction ID errors from MCIn are indicated</p>
0	31:28	r	<p><b>Reserved</b>            Read as 0; should be written with 0.</p>

### Bridge Control Register

The layout of this register is different for each instance. Consult the product manual for the effective layouts.

## On-Chip System Connectivity {and Bridges}

### BRCON

#### Bridge Control Register

(00430<sub>H</sub>)Application Reset Value: XXXX XXXX<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0							
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		0		0		1		0		0		0		OLDAEN	
rw		r		rw	rw	rw	r		rw		r		r		rw

Field	Bits	Type	Description
OLDAEN	0	rw	<b>Online Data Acquisition Enable</b> This bit is used to control trap generated for write accesses to the OLDA address range associated with this domain.  $0_B$ Trap generated on a write access to the OLDA memory range. $1_B$ No trap generated on a write access to the OLDA memory range.
0	5:1, 8:7, 12:11, 31:20	r	<b>Reserved</b> Read as 0; shall be written with 0.
0	6, 10, 19:13	rw	<b>Reserved</b> Read as 0; shall be written with 0.
1	9	rw	<b>Reserved</b> Read as 1; shall be written with 1.

### Access Enable Register 0

The Access Enable Register 0 controls access for transactions with the on chip bus master TAG ID 0 to 31 (see On Chip Bus chapter for the TAG ID <-> master mapping). The registers ACCEN0 / ACCEN1 provide one enable bit for each possible 6-bit TAG ID. The mapping of TAG IDs to ACCEN0.ENx is: EN0 -> TAG ID 0, EN1 -> TAG ID 1, ... EN31 -> TAG ID 31. For modules connected to SRI bus, ACCEN0 register controls write accesses from all on chip bus masters. Read access is not controlled.

### ACCENO

#### Access Enable Register 0

(004F0<sub>H</sub>)Application Reset Value: FFFF FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
EN15								EN0							
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rw															

## On-Chip System Connectivity {and Bridges}

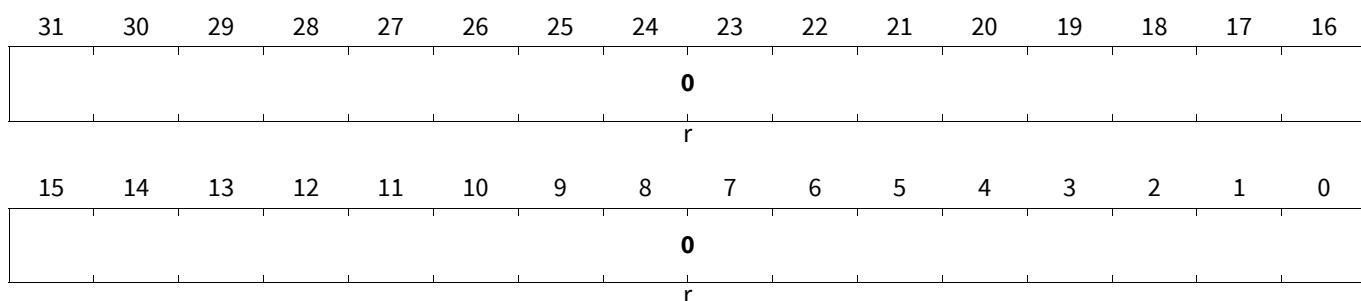
Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<p><b>Access Enable for Master TAG ID n</b></p> <p>This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n:</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> Write access will not be executed</li> <li>1<sub>B</sub> Write access will be executed</li> </ul>

### Access Enable Register 1

The Access Enable Register 1 controls access for transactions with the on chip bus master TAG ID 32 to 63 (see On Chip Bus chapter for the TAG ID <-> master mapping). ACCEN1 is not implemented with register bits, as the related TAG IDs belong to masters that cannot access the SRI SFR's in AURIX™ devices. All writes from master TAG ID's > 32 will result in the write access not to be executed. The mapping of TAG IDs to ACCEN1.ENx is: EN0 -> TAG ID 32, EN1 -> TAG ID 33, ... EN31 -> TAG ID 63

#### ACCEN1

**Access Enable Register 1** (004F8<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>0</b>	31:0	r	<p><b>Access Enable Register 1</b></p> <p>Read as 0; should be written with 0. Write access will not be executed</p>

## On-Chip System Connectivity {and Bridges}

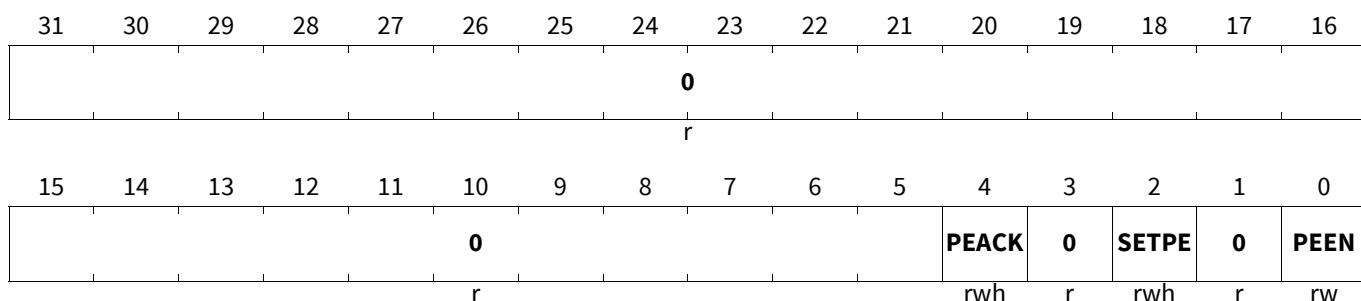
### 4.4.2 SCI Control Registers

Note: Only the registers assigned to implemented SCIs are present. Registers for unimplemented SCIs are treated as reserved. See [Chapter 4.3.4](#) for which SCIs are implemented.

#### Protocol Error Control Register x

##### PECONx (x=0-15)

**Protocol Error Control Register x**       $(00000_H + x*20_H)$       **Application Reset Value: 0000 0001<sub>H</sub>**



Field	Bits	Type	Description
PEEN	0	rw	<b>Protocol Error Enable</b> 0 <sub>B</sub> Protocol errors are not indicated and no information is captured. 1 <sub>B</sub> Protocol errors are indicated and information is captured.
SETPE	2	rwh	<b>Set Protocol Error</b> This allows SW to mimic a protocol error and present an indication similar to a hardware detected error. After setting this bit, it is automatically cleared by the hardware in the cycle after the write. 0 <sub>B</sub> No protocol error indication is generated 1 <sub>B</sub> A protocol error indication is generated
PEACK	4	rwh	<b>Protocol Error Acknowledge</b> Writing a one to this bit while it's set has the following results: The error lock of the corresponding ERRADDRx and ERRx are released, and the registers will be updated for the next protocol error detected (see <a href="#">Chapter 4.3.3.5</a> ). After setting this bit, it is automatically cleared by the hardware in the cycle after the write. 0 <sub>B</sub> Default value 1 <sub>B</sub> A Protocol Error for this arbiter has been indicated. The corresponding ERRADDR and ERR registers are not updated for new errors.
0	1, 3, 31:5	r	<b>Reserved</b> Read as 0; should be written with 0.

#### SCIx Arbiter Priority Register

Identical functionality can be achieved by setting all priorities to high compared with setting all priorities to low. HPRS only affects arbitration when high priority requests arrive so frequently that there are no arbitration rounds with only low priority requests. When this occurs (no high priority free arbitration rounds), a maximum of HPRS

## On-Chip System Connectivity {and Bridges}

grants are given to high priority masters, and then a single grant is given to the low priority round robin winner. See [Table 65](#) for effects on latency on different numbers of high priority requesters and different values of HPRS.

**Note:** Only the MCIn\_P bits assigned to implemented MCIs are present. Bits corresponding to unimplemented MCIs are treated as reserved bits. See [Chapter 4.3.4](#) for implemented MCI per domain.

3. Reset values for bits/bitfields coupled to masters or slaves that are not configured or enabled are zero.
4. The reset value for SRI0.PRIORITYx ( $x=0..19$ ) is  $0007\ 0043_H$ ; SRI1.PRIORITYx ( $x=0..5$ ) is  $0007\ 0002_H$ ; and SRI2.PRIORITYx ( $x=0..4$ ) is  $0007\ 0000_H$ .

### PRIORITYx ( $x=0-15$ )

<b>SCIx Arbiter Priority Register</b>																<b>(00008<sub>H</sub>+x*20<sub>H</sub>)</b>		<b>Application Reset Value: 0007 0XXX<sub>H</sub></b>			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	0	rw				
																		rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	0	rw	rw	rw	rw	rw	rw
				MCI11_P	MCI10_P	MCI9_P	MCI8_P	MCI7_P	MCI6_P	MCI5_P	MCI4_P	MCI3_P	MCI2_P	MCI1_P	MCI0_P		rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MCIn_P (n=0-11)	n	rw	<b>MCIn Priority</b> 0 <sub>B</sub> MCIn requests are arbitrated in the low priority round robin. 1 <sub>B</sub> MCIn requests are arbitrated in the high priority round robin.
<b>HPRS</b>	18:16	rw	<b>High Priority Round Share</b> Number of transactions to give to the high priority round robin before a transaction from low priority round (when request saturated). This number may not be less than the number of high priority MCI programmed via. MCIn_P.
<b>0</b>	15:12, 31:19	r	<b>Reserved</b> Read as 0; should be written with 0.

### SCI x Error Address Capture Register

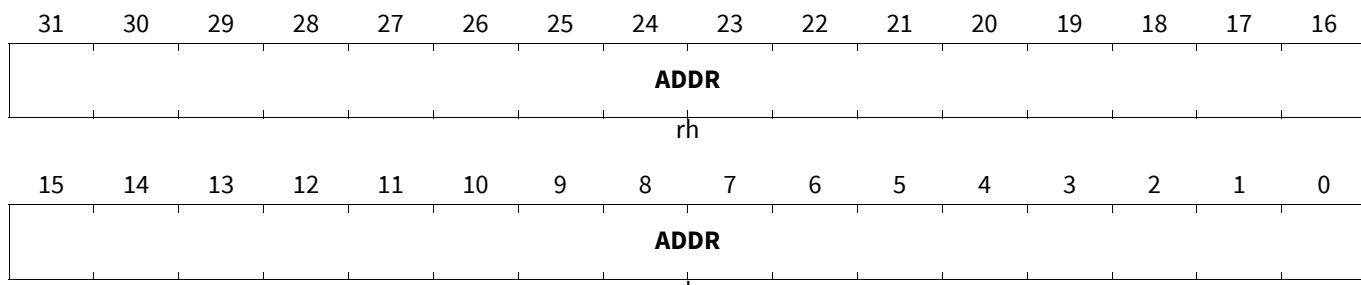
#### Notes

1. The default value can differ from the one shown here, because a constant can be used to reduce the number of compared bits in the arbitration if a slave occupies only a limited address area. For more details, see the design specification of the SRI crossbar.

## On-Chip System Connectivity {and Bridges}

### ERRADDRx (x=0-15)

**SCI x Error Address Capture Register** **(00010<sub>H</sub>+x\*20<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**

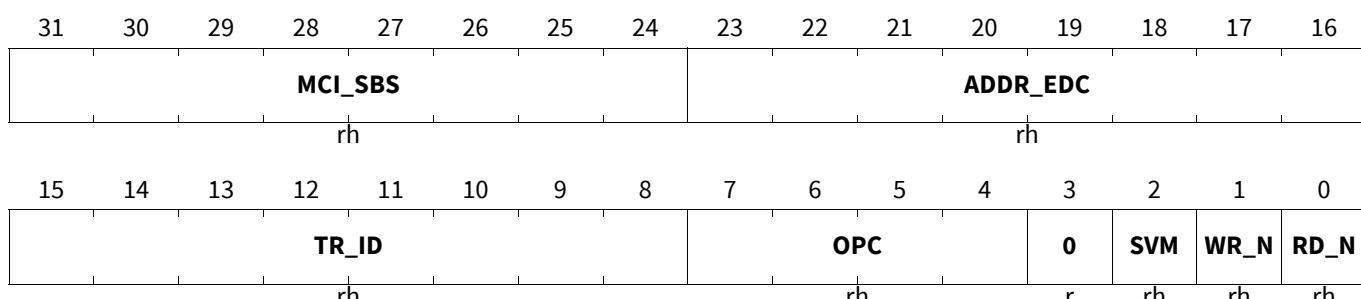


Field	Bits	Type	Description
<b>ADDR</b>	31:0	rh	<b>Transaction Address</b> This bitfield contains the address of the erroneous transaction from the address phase

### SCI x Error Capture Register

### ERRx (x=0-15)

**SCI x Error Capture Register** **(00018<sub>H</sub>+x\*20<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>RD_N</b>	0	rh	<b>Read Status - RD</b> $0_B$ The read signal line was asserted (read or start of RMW transaction) $1_B$ The read line was deasserted (no read transaction)
<b>WR_N</b>	1	rh	<b>Write Status - WR</b> $0_B$ The write signal line was asserted (write or start of RMW transaction) $1_B$ The write signal line was deasserted (no write transaction)
<b>SVM</b>	2	rh	<b>Supervisor Mode Status</b> $0_B$ The supervisor mode signal line was deasserted $1_B$ The supervisor mode signal line was asserted
<b>OPC</b>	7:4	rh	<b>Operation Code</b> This field contains the opcode of the erroneous transaction; see <a href="#">Table 68</a> for details.

## On-Chip System Connectivity {and Bridges}

Field	Bits	Type	Description
<b>TR_ID</b>	15:8	rh	<b>Transaction ID</b> This field contains the Master's transaction ID of the erroneous transaction. The Transaction ID is built out of the Master's 6-bit unique TAG ID (TR_ID[5:0]), and a 2-bit running number TR_ID[7:6].
<b>ADDR_EDC</b>	23:16	rh	<b>Address Phase Error Detection Information</b> This field contains the Address Phase Error Detection Information of the erroneous transaction.
<b>MCI_SBS</b>	31:24	rh	<b>MCI Sideband Signals [7:0]</b> This bit field contains the MCI Sideband Signals [7:0] that are related to the transaction information captured. In the AURIX™ family, the sideband signals are used by the DMA to provide information about the DMA requestor of a DMA transaction (for the encoding see <a href="#">Table 66</a> ).
<b>0</b>	3	r	<b>Reserved</b> Read as 0; should be written with 0.

### OPC encoding

The OPC field contains information about the type of SRI transaction for which information has been captured. For single data transactions, this field indicates the number of bytes. For block transfer, this field indicates the expected number of data beats.

**Table 68 Operation Code Encoding**

OPC[3:0]	Identifier	Description
0000	SDTB	Single Data Transfer Byte(8 bit)
0001	SDTH	Single Data Transfer Half-Word(16 bit)
0010	SDTW	Single Data Transfer Word(32 bit)
0011	SDTD	Single Data Transfer Double-Word(64 bit)
0100-0111	-	Reserved
1000	BTR2	Block Transfer Request (2 transfers) Wrap Around Addresses are used.
1001	BTR4	Block Transfer Request (4 transfers) Wrap Around Addresses are used.
1010-1111	-	Reserved

## On-Chip System Connectivity {and Bridges}

### 4.5 S2S Bridge

The S2S bridge is unidirectional, but bidirectional functionality is achieved where required by placing two bridges in opposite directions between two SRI domains. S2S bridges transmit all SRI transaction types transparently (no master tag or address modification). They are designed to minimize latency between the two domains connected by the bridge.

#### Error Behavior

The S2S bridge is special in its handling of errors. It transmits most errors and does not report them itself.

#### 4.5.1 EDC Errors

Address EDC errors are passed from the initiating domain and left to the receiving slave, to interrupt or raise alarms as configured.

Read data EDC errors are transferred from the slave domain to the initiating master, to interrupt or raise alarms as configured.

Write data EDC errors will be transferred from the initiating domain, and left to the receiving slave to interrupt or raise alarms as configured.

#### 4.5.2 Protocol Errors

SRI protocol errors on reads are transferred from the slave domain to the initiating master, to interrupt or raise alarms as configured.

As write transactions are buffered, SRI protocol errors will not be signalled in the initiating SRI domain by the S2S bridge, as it has no knowledge of the error condition. In the receiving domain, protocol errors will be left to the receiving slave, to interrupt or raise alarms as configured.

#### 4.5.3 Transaction ID Errors

Transaction ID errors on reads are transferred from the slave domain to the initiating master, to interrupt or raise alarms as configured.

Transaction ID errors on writes are transferred from the master domain to the receiving slave, to interrupt or raise alarms as configured.

#### 4.5.4 Transaction Errors for Writes via S2S Bridge

Write transactions from one SRI domain to another SRI domain, resources are handled as posted writes. This means that a write operation from a SRI Master through the S2S Bridge can be finished on the source SRI Fabric, and the S2S autonomously completes the write later on the destination SRI Fabric. If the S2S write transaction results in an error on the destination domain, the error information is not passed back to the source SRI Fabric.

Note that this behavior occurs only for write operations via the S2S Bridge. It can also be triggered by the write cycle of a read-modify-write transaction that causes an error on the destination SRI domain.

## On-Chip System Connectivity {and Bridges}

### 4.6 SFI\_F2S Bridge

This section describes the functionality of the SFI\_S2S Bridge, a bridge from the SPB to the SRI Fabric.

#### 4.6.1 Functional Overview

The SFI\_S2S Bridge implements a uni-directional bus bridge that forwards transactions from an implementation of an FPI protocol bus, in this instance the System Peripheral Bus (SPB), to the SRI Fabric. The bridge supports all FPI transactions on the SPB and those transactions of the SRI Fabric required to implement them.

The bridge is transparent for the address of a transaction and the master TAG of the SPB master. However, transactions are mapped as described in [Table 69](#). For performance reasons, write transactions from SPB masters to SRI resources are handled as posted writes. The SFI\_F2S bridge is able to buffer multiple posted writes.

**Table 69 FPI Transaction to SRI Transaction Mapping**

FPI Transaction (32-bit data bus)	SRI Transaction (64-bit data bus)
Single Byte Transfer(8-bit)	Single Data Transfer Byte(8 bit)
Single Half-Word Transfer(16-bit)	Single Data Transfer Half-Word(16 bit)
Single Word Transfer(32-bit)	Single Data Transfer Word(32 bit)
2-Word Block Transfer	Single Data Transfer Double-Word(64 bit)
4-Word Block Transfer (aligned)	Block Transfer Request (2 transfers)
8-Word Block Transfer (aligned)	Block Transfer Request (4 transfers).

#### Transaction Errors for Writes via the SFI\_F2S Bridge

Write transactions from SPB masters to SRI resources are handled as posted writes. This means that a write operation from a SPB master through the SFI\_F2S Bridge can finish on the SPB, and the SFI\_F2S autonomously completes the write later on the SRI. If the SRI write transaction results in an error, the error information is not passed back to the SPB bus. The error condition will be left to the receiving SRI slave to interrupt or raise alarms as configured.

Note that this behavior occurs only for write operations via the SFI\_F2S Bridge. It can also be triggered by the write cycle of a read-modify-write transaction which causes an error on the SRI Fabric.

## On-Chip System Connectivity {and Bridges}

### 4.7 SFI\_S2F Bridge

This section describes the functionality of the SFI\_S2F Bridge, a bridge from the SRI Fabric to the BBB.

#### 4.7.1 Functional Overview

The SFI\_S2F Bridge is implemented as an uni-directional bus bridge that forwards transactions from the SRI Fabric to the Back Bone Bus (BBB). The bridge supports all SRI transactions on the SRI Fabricm, and those transactions of the FPI Bus required to implement them.

The bridge is transparent for the address of a transaction and the master TAG of the SRI master. However, transactions are mapped as described in **Table 70**. For performance reasons, write transactions from the SRI to BBB resources are handled as posted writes. The SFI\_S2F bridge is able to buffer multiple posted writes.

**Table 70 SRI Transaction to FPI Transaction Mapping**

SRI Transaction (64-bit data bus)	FPI Transaction (32-bit data bus)
Single Data Transfer Byte(8 bit)	Single Byte Transfer (8-bit)
Single Data Transfer Half-Word(16 bit)	Single Half-Word Transfer (16-bit)
Single Data Transfer Word(32 bit)	Single Word Transfer (32-bit)
Single Data Transfer Double-Word(64 bit)	2-Word Block Transfer (aligned)
Block Transfer Request (2 transfers)	4-Word Block Transfer
Block Transfer Request (4 transfers).	8-Word Block Transfer

#### Transaction Errors for Writes via the SFI\_S2F Bridge

Write transactions from SRI Masters to BBB resources are handled as posted writes. This means that a write operation from a SRI Master through the SFI\_S2F Bridge can be finished on the SRI Fabric, and the SFI\_S2F autonomously completes the write later on the BBB. If the BBB write transaction results in an error on the BBB, the error information is not passed back to the SRI Fabric. The error condition is detected by the control logic of the BBB (BCU on the BBB), to interrupt or raise alarms as configured.

Note that this behavior occurs only for write operations via the SFI\_S2F Bridge. It can also be triggered by the write cycle of a read-modify-write transaction that causes an error on the BBB.

### 4.8 Resource Access Times

These tables describe the CPU access times to various resources in CPU clock cycles for the AURIX™ TC3xx Platform. In the case of load or fetch accesses, the access times are the minimum number of CPU stall cycles to complete the access. If there is a conflict for the resource accessed, there may be additional stall cycles till the conflicting access completes.

For write access, the access times are the maximum for a sequence of such access (non-conflicting). In many cases for a singleton access, or a short sequence, write buffering reduces the stall effect seen by a CPU, sometimes to 0. However, as with loads and fetches, if there is a conflict for the resource accessed, there may be additional stall cycles till the conflicting access completes.

## On-Chip System Connectivity {and Bridges}

**Table 71 Access latency for global resources**

CPU Access Type	CPU stall cycles
Data read from System Peripheral Bus (SPB) <sup>1)</sup>	(4 + Module Wait State <sup>2)</sup> ) ( $f_{CPU} = f_{SPB}$ ) 2 * (4 + Module Wait State <sup>2)</sup> ) ( $f_{CPU} = 2*f_{SPB}$ ) 3 * (4 + Module Wait State <sup>2)</sup> ) ( $f_{CPU} = 3*f_{SPB}$ )
Data write to System Peripheral Bus (SPB) <sup>1)</sup>	(4 +Module Wait State <sup>2)</sup> ) ( $f_{CPU} = f_{SPB}$ ) 2 * (4 + Module Wait State <sup>2)</sup> ) ( $f_{CPU} = 2*f_{SPB}$ ) 3 * (4 + Module Wait State <sup>2)</sup> ) ( $f_{CPU} = 3*f_{SPB}$ )
Data read from Back Bone Bus (BBB) <sup>3)</sup> (TC39x, TC37xED)	9 + (5 + Module Wait State <sup>2)</sup> ) ( $f_{CPU} = f_{BBB}$ ) 9 + 2 * (5 + Module Wait State <sup>2)</sup> ) ( $f_{CPU} = 2*f_{BBB}$ )
Data write to Back Bone Bus (BBB) <sup>3)</sup> (TC39x, TC37xED)	5 + (4 + Module Wait State <sup>2)</sup> ) ( $f_{CPU} = f_{BBB}$ ) 5 + 2 * (4+ Module Wait State <sup>2)</sup> ) ( $f_{CPU} = 2*f_{BBB}$ )
Data read from Back Bone Bus (BBB) (TC35x, TC33xED)	6 + (5 + Module Wait State <sup>2)</sup> ) ( $f_{CPU} = f_{BBB}$ ) 6 + 2 * (5 + Module Wait State <sup>2)</sup> ) ( $f_{CPU} = 2*f_{BBB}$ )
Data write to Back Bone Bus (BBB) (TC35x, TC33xED)	3 + (4 + Module Wait State <sup>2)</sup> ) ( $f_{CPU} = f_{BBB}$ ) 3 + 2 * (4 + Module Wait State <sup>2)</sup> ) ( $f_{CPU} = 2*f_{BBB}$ )

- 1) The final number of stall cycles will depend on the real number of WS generated by the target resource.
- 2) The number of wait states for read and for write accesses is  $\geq 1$  and depends on the accessed module and its configuration.
- 3) When SFI\_S2F is connected to XBar2 (TC39x and TC37xED) there is an additional latency due to access going through an S2S.

### Notes on System Peripheral Bus

Access to critical module registers is implemented with 1 Wait State (e.g. QSPI, ASCLIN, EVADC and DSADC result registers, DMA). Other modules are partially implemented with  $> 1$  Wait State where a single wait state implementation was not possible.

Additional Wait States due to:

- Access to module internal SRAMs (e.g. MCAN, DMA, ERAY, GTM)
- System infrastructure modules (e.g. MTU, STM)
- Access to module registers mapped to a different clock domain (e.g. STM, GTM)
- Access to control registers in large and complex modules that require internal registers stages to meet the timing (e.g. GTM)

**Table 72 CPU Accesses: Stall cycles for local and SRI resources**

	Local CPU	Local SRI	Remote SRI Domain <sup>1)</sup>
Data read from DSPR	0	7	10
Data write to DSPR	0	5, 3 <sup>2)</sup>	5, 4 <sup>2)</sup>
Instruction fetch from DSPR	See local SRI column <sup>3)</sup>	7	10
Data read from DLMU	0	7	10
Data write to DLMU	2	5, 3 <sup>2)</sup>	5, 4 <sup>2)</sup>
Instruction fetch from DLMU	See local SRI column <sup>3)</sup>	7	10
Data read from PSPR	See local SRI column <sup>3)</sup>	7	10
Data write to PSPR	See local SRI column <sup>3)</sup>	5, 3 <sup>2)</sup>	5, 4 <sup>2)</sup>

## On-Chip System Connectivity {and Bridges}

**Table 72 CPU Accesses: Stall cycles for local and SRI resources**

	Local CPU	Local SRI	Remote SRI Domain <sup>1)</sup>
Instruction fetch from PSPR	0	7	10
Data read from PFlash	5 + PWS <sup>4)</sup>	10 + PWS <sup>4)</sup>	13 + PWS <sup>4)</sup>
Instruction fetch from PFlash (buffer miss)	2 + PWS <sup>5)</sup>	9 + PWS <sup>5)</sup>	12 + PWS <sup>5)</sup>
Instruction fetch from PFlash (buffer hit)	3	6	9
Data read from LMU	n.a.	7	10
Data write to LMU	n.a.	5, 3 <sup>2)</sup>	5, 4 <sup>2)</sup>
Instruction fetch from LMU	n.a.	7	10
Data read from DFlash	n.a.	5 + 3*(3+DCWS) <sup>6)</sup>	8 + 3*(3+DCWS) <sup>6)</sup>
DFlash runs on FSI clock. $f_{CPU} = 3*f_{FSI}$			
Data read access from EMEM (TC39x, TC37xED)	n.a.	n.a.	14, 15 <sup>7)</sup>
Data write access to EMEM (TC39x, TC37xED)	n.a.	n.a.	9
Data read access from EMEM (TC35x, TC33xED)	n.a.	11, 12 <sup>7)</sup>	n.a.
Data write access to EMEM (TC35x, TC33xED)	n.a.	9	n.a.
Data read access from EMEM (TC3Ax)	n.a.	11, 12 <sup>7)8)</sup>	14, 15 <sup>7)9)</sup>
Data write access to EMEM (TC3Ax)	n.a.	9 <sup>8)</sup>	9 <sup>9)</sup>
Data read access from DAM	n.a.	10	13
Data write access to DAM	n.a.	7	7

- 1) Only applies to products with SRI extenders. Additional latency due to access going through an S2S
- 2) With pipelining
- 3) Data access to code side memories or Code accesses to data side memories are made via the SRI bus
- 4) PWS: Configured PFlash Wait States (Includes cycles for PFlash access cycles only). ECC correction latency is only incurred when the incoming data requires ECC correction.
- 5) PWS: Configured PFlash Wait States (Includes cycles for PFlash access cycles only). ECC correction latency is only incurred when the incoming data requires ECC correction.
- 6) DCWS: Configured DFlash Corrected Wait States (Includes cycles for DFlash access cycles and ECC correction latency)
- 7) The EMEM works on  $f_{BBB}$  clock which is lower than the  $f_{SRI}$  there could be one additional synchronisation cycle for the request to be acknowledged by the EMEM.
- 8) EMEM4 and EMEM5 are located on domain 0
- 9) EMEM0/EMEM1/EMEM2/EMEM3 are located on domain 2

### Notes on PFlash and DFlash accesses

The PFlash waitstates are described in the NVM section and their configuration is done via the HF\_PWAIT register. The DFlash waitsates are described in the NVM section and their configuration is done via the HF\_DWAIT register. In both cases the values programmed in the registers are 1 less than the actual value. (e.g. HF\_DWAIT.RFLASH=9 corresponds to 10 read cycles, and similarly HF\_DWAIT.ECC=1 corresponds to 2 ECC cycles).

e.g: Number of Stall cycles for HF\_DWAIT.RFLASH=9 and HF\_DWAIT.ECC=1 accessed on the local SRI

$$\begin{aligned}
 \text{DCWS} &= (\text{DFLASH READ CYCLES}) + (\text{DFLASH ECC CYCLES}) \\
 &= (\text{HW_DWAIT.RFLASH} + 1) + (\text{HW_DWAIT.ECC} + 1) \\
 &= 12
 \end{aligned}$$

$$\text{Stall Cycles} = 5 + 3 * (3 + \text{DCWS})$$

## On-Chip System Connectivity {and Bridges}

= 50

### 4.9 Revision History

**Table 73 Revision History**

Reference	Changes to Previous Version	Comment
<b>V1.1.13</b>		
<a href="#">Page 22</a>	Updated BRCON bitfield to show that bit 9 while reserved will read as 1 and should be updated to 1	
<a href="#">Page 15</a>	Updated TC33xEXT configuration in SRI Master Domain 0 table to show 3 DMA MIF's.	
<a href="#">Page 32</a>	Corrected Data read from PFlash stall cycles. The table was incorrectly using PCWS but should have used PWS since the penalty for ECC is only incurred if an ECC correction is required and not for all accesses.	
<a href="#">Page 33</a>	Updated footnote regarding data read access to PFlash to indicate that ECC stall penalty is only incurred if and ECC correction is required	
<a href="#">Page 33</a>	Updated footnote regarding instruction fetch from PFlash (buffer miss) to indicate that ECC stall penalty is only incurred if and ECC correction is required	
<b>V1.1.14</b>		
	No change.	
<b>V1.1.15</b>		
<a href="#">Page 15</a>	Renamed TC38xEXT in TC3Ex in SRI Domain 0 Master mapping table.	
<a href="#">Page 15</a>	Added new TC3Ax derivative in SRI Domain 0 Master mapping table.	
<a href="#">Page 16</a>	Renamed TC38xEXT in TC3Ex and added new TC3Ax derivative in SRI Domain 0 Slave mapping table.	
<a href="#">Page 16</a>	Added new TC3Ax derivative in SRI Domain 0 Master mapping table.	
<a href="#">Page 17</a>	Added new TC3Ax derivative in SRI Domain 2 Master mapping table.	
<a href="#">Page 18</a>	Added new TC3Ax derivative in SRI Domain 2 Slave mapping table.	
<a href="#">Page 32</a>	Added new TC3Ax derivative in CPU Stalls table	
<b>V1.1.16</b>		
<a href="#">Page 15</a>	Updated SRI Domain 0 Master mapping table: Moved HSSL1 to Domain 0. Removed 'S2S2 D2D0'	
<a href="#">Page 17</a>	Updated SRI Domain 2 Master mapping table: Moved HSSL1 to Domain 0.	
<a href="#">Page 18</a>	Updated SRI Domain 2 Slave mapping table: Moved HSSL1 to Domain 0.	

## 4.10 FPI Interconnect

The System on Chip communication is based on two On Chip Bus protocols:

- Shared Resource Interconnect (SRI, Cross Bar based interconnect, 64 bit data bus)
- Fast Peripheral Interconnect (FPI, Multi-Master interconnect, 32 bit data bus)

The SRI connects the TC1.6 CPUs, the main high bandwidth peripherals and the DMA module to its local resources for instruction fetches and data accesses.

The FPI connects the high speed peripherals, e.g. TC1.6 CPUs, DMA, to the medium and low bandwidth peripherals.

### Scope of this Document

This document is valid for the TC3xx and covers the topics:

- FPI Bus architecture
- FPI Bus instances (SPB, BBB)
- FPI Bus Operations

### 4.10.1 Feature List

The FPI Bus interconnects the on-chip peripheral functional units with the processor subsystem.

The FPI Bus is designed to be quick to be acquired by on-chip functional units, and quick to transfer data. The low setup overhead of the FPI Bus access protocol guarantees fast FPI Bus acquisition, which is required for time-critical applications.

The FPI Bus is designed to sustain high transfer rates. Pipelining of transfers is supported by the parallel handling of request, address and data phases of transfers which allow the FPI Bus to operate close to its peak bandwidth.

Additional features of the FPI Bus include:

- Optimized for high speed and high performance
- Support of multiple bus masters and pipelined transactions
- 32-bit wide address and data buses
- 8-, 16-, and 32-bit data transfers
- 64-, 128-, and 256-bit block transfers
- Slave-controlled wait state insertion
- Timeout detection and handling
- Support of atomic CPU operations e.g. LDMST, ST.T, SWAP.W, CMPSWP and SWPMSK
- Flexible arbitration schemes (priority, one round robin group, starvation prevention)
- Starvation prevention mechanism that can take care that even low priority requests will be granted after a configurable number of arbitration cycles, permanently enabled
- Default slave (takes over transactions to not implemented address)
- Debug support (captures transaction information in case of a timeout or bus error)
- Address Phase includes Master TAG ID and Supervisor Mode information
- All slave modules implemented with a TAG ID based access protection that provides a generic write protection for the control registers (SPB/BBB: write protection only)
- Information integrity support covering SPB/BBB address phase signals, transmitted read/write data and control signals
- SPB/BBB Address Phase includes Supervisor Mode information (covered by the FPI information integrity support)

The functional units of the device are connected to the FPI Bus via FPI Bus interfaces. An FPI Bus interfaces acts as bus agents, requesting bus transactions on behalf of their functional unit, or responding to bus transaction requests.

There are two types of bus agents:

- FPI Bus master agents can initiate FPI Bus transactions and can also act as slaves.
- FPI Bus slave agents can only react and respond to FPI Bus transaction requests in order to read or write internal registers of slave modules as for example memories.

When an FPI Bus master attempts to initiate a transfer on the FPI Bus, it first signals a request for bus ownership to the bus control unit (SBCU). When bus ownership is granted by the SBCU, an FPI Bus read or write transaction is initiated. The unit targeted by the transaction becomes the FPI Bus slave, and responds with the requested action.

Some functional units operate only as slaves, while others can operate as either masters or slaves on the FPI Bus.

FPI Bus arbitration is performed by the Bus Control Unit (SBCU) of the FPI Bus. In case of bus errors, the SBCU generates an interrupt request to the CPU and provides debugging information about the actual bus error to the CPU.

#### 4.10.1.1 Delta to TC2xx

Major differences of the AURIX™ TC3xx Platform Bus System architecture compared to previous TC2xx products:

- Assignment of TAG IDs to on chip bus master resources was changed (see [On Chip Bus Master TAG Assignments](#))
- Adapted mapping of SPB master interfaces SBCU\_DBGRNT and SBCU\_DBGNTT register bits.
- FPI EDC: Introduced End to End Error Detection (EDC) mechanism ([FPI Bus Integrity Support](#))
- FPI EDC: Introduced BCU control registers for the EDC mechanism (see also SBCU and EBCU Registers in the Appendix document, chapter BCU)
  - BCU\_ALSTAT0, EDC Alarm Status Register
  - BCU\_ALCLR $x$  ( $x=0-3$ ), EDC Alarm Clear Register
  - BCU\_ALCTRL, EDC Control Register
  - BCU\_FEGEN, FPI Error Generation Register
- Adapted priority mapping of SPB master interfaces (BCU\_PRIOH and BCU\_PRIOL registers))
- FPI arbitration algorithm: Added Round Robin group to Priority 8 ([“FPI Bus Arbitration” on Page 38](#))
- FPI arbitration: Adapted the default configuration of the FPI master priorities
- Back Bone Bus (BBB): the description of the Back Bone Bus BCU (EBCU) was added

#### 4.10.2 Overview

The AURIX™ TC3xx Platform has up to two FPI Bus instances:

- System Peripheral Bus (SPB): Main non-ADAS system and communication peripherals
- Back Bone Bus (BBB): Emulation Device related and ADAS related peripherals, available in ADAS / Emulation Devices only

This section gives an overview of the on-chip FPI Bus instances (SPB, BBB). It describes its Bus Control Units (SBCU, EBCU), the bus characteristics, bus arbitration, scheduling, prioritizing, error conditions, and debugging support.

#### 4.10.2.1 Bus Transaction Types

This section describes the SPB transaction types.

## Single Transfers

Single transfers are byte, half-word, and word transactions that target any slave connected to SPB. Note that the SFI Bridge operates as an SPB master.

## Block Transfers

Block transfers operate in principle in the same way as single transfers do, but one address phase is followed by multiple data phases. Block transfers can be composed of 2 word, 4 word, or 8 word transfers.

**Note:** *In general, block transfers (2 word, 4 word, or 8 word) cannot be executed in the AURIX™ TC3xx Platform with peripheral units that operate as FPI Bus slaves during an FPI Bus transaction.*

Block transfers are initiated by the following CPU instructions: LD.D, LD.DA, MOV.D, ST.D and ST.DA. Additionally there are communication peripherals that are able to generate block transfers (e.g. Ethernet).

## Atomic Transfers

Atomic transfers are generated by LDMST, ST.T and SWAP.W instructions that require two single transfers. The read and write transfer of an atomic transfer are always locked and cannot be interrupted by another bus masters. Atomic transfers are also referenced as read-modify-write transfers.

**Note:** *See also [Table 75](#) for available FPI Bus transfer types.*

### 4.10.2.2 Reaction of a Busy Slave

If an FPI Bus slave is busy at an incoming FPI Bus transaction request, it can delay the execution of the FPI Bus transaction. The requesting FPI Bus master releases the FPI Bus for one cycle after the FPI Bus transaction request, in order to allow the FPI Bus slave to indicate if it is ready to handle the requested FPI Bus transaction. This sequence is repeated as long as the slave indicates that it is busy.

**Note:** *For the FPI Bus default master, the one cycle gap does not result in a performance loss because it is granted the FPI Bus in this cycle as default master if no other master requests the FPI Bus for some other reasons.*

### 4.10.2.3 Address Alignment Rules

FPI Bus address generation is compliant with the following rules:

- Half-word transactions must have a half-word aligned address ( $A_0 = 0$ ). Half-word accesses on byte lanes 1 and 2 addresses are illegal.
- Word transactions must always have word-aligned addresses ( $A[1:0] = 00_B$ ).
- Block transactions must always have block-type aligned addresses.

### 4.10.2.4 FPI Bus Basic Operations

This section describes some basic transactions on the FPI Bus.

The example in [Figure 36](#) shows the three cycles of an FPI Bus operation:

1. **Request/Grant Cycle:** The FPI Bus master attempts to perform a read or write transfer and requests for the FPI Bus. If the FPI Bus is available, it is granted in the same cycle by the FPI Bus controller.
2. **Address Cycle:** After the request/grant cycle, the master puts the address on the FPI Bus, and all FPI Bus slave devices check whether they are addressed for the following data cycle.

3. **Data Cycle:** In the data cycle, either the master puts write data on the FPI Bus which is read by the FPI Bus slave (write cycle) or vice versa (read cycle).

Transfers 2 and 3 show the conflict when two master try to use the FPI Bus and how the conflict is resolved. In the example, the FPI Bus master of transfer 2 has a higher priority than the FPI Bus master of transfer 3.

Bus Cycle	1	2	3	4	5	
Transfer 1	Request/ Grant	Address Cycle	Data Cycle			
Transfer 2		Request/ Grant	Address Cycle	Data Cycle		
Transfer 3			Request/Grant	Address Cycle	Data Cycle	

MCA06109

**Figure 36 Basic FPI Bus Transactions**

At a block transfer, the address cycle of a second transfer is extended until the data cycles of the block transfer are finished. In the example of [Figure 37](#), transfer 1 is a block transfer, while transfer 2 is a single transfer.

Bus Cycle	1	2	3	4	5	6	7	
Transfer 1	Request/ Grant	Address Cycle	Data Cycle	Data Cycle	Data Cycle	Data Cycle		
Transfer 2		Request/ Grant					Address Cycle	Data Cycle

MCA06110

**Figure 37 FPI Bus Block Transactions**

#### 4.10.3 Functional Description (SBCU, EBCU)

The AURIX™ TC3xx Platform incorporates up to two FPI Bus Control Units:

- SBCU on the System Peripheral Bus (SPB)
- EBCU on the Back Bone Bus (BBB), available in ADAS / Emulation devices only

##### 4.10.3.1 FPI Bus Arbitration

The arbitration unit of the BCU arbitrates among the master interfaces that are requesting for bus access. The priority for each master is defined in the registers PRIOH and PRIOL. During arbitration, the bus is granted to the requesting master with the highest priority. If no request is pending, the bus is granted to a default master. If no bus master takes the bus, the BCU itself will set the FPI Bus into an idle state. The BCU arbitrates always at the end of the current Address Phase and decides which master agent shall be granted.

The mapping of implemented master agents to the PRIOH / PRIOL register is described in the Appendix document, chapter BCU.

The arbitration is done at the end of a transaction

#### 4.10.3.1.1 Arbitration on the System Peripheral Bus

The AURIX™ TC3xx Platform has multiple bus master connected to the SPB

- Each master is assigned to 4-bit priority bit field in the PRIOH or in the PRIOL register
- The value in a priority bit field defines the related SPB master priority used for the bus arbitration
- A lower priority number has a higher priority, 0 is the highest priority

#### 4.10.3.1.2 Default Master

Any FPI master must be able to act as default master. If no master is requesting for bus access, the FPI master that was most recently granted will be granted to the Default Master. After reset, the master with the priority 0 will be the Default Master.

#### 4.10.3.1.3 Arbitration Algorithms

The arbitration algorithm implemented in the BCU combines three layers of arbitration

- Priority driven arbitration of master agents with a pending request (Priority 0 - 15, 0 = highest priority)
- Round Robin arbitration for a group of up to 8 master on Priority 8.
- Starvation Prevention mechanism that can increase master priorities during Starvation Prevention process

The default priority of each connected FPI master agent can be changed during runtime via its related bit field in the registers PRIOH and PRIOL (see 'Changing Master Priorities').

It must be ensured that two FPI master agents don't have the same priority, with the exception of Priority 8 (round robin group priority).

If more than one master is configured with the same priority (but not with the round robin priority 8) the arbitration will still work but the sequence how these master will be granted can not be guaranteed.

##### Priority Driven Arbitration

The general arbitration algorithm is priority driven where priority 0 is the highest priority and 15 the lowest one. If multiple masters are requesting, the master with the highest priority will win the next arbitration round (see also starvation prevention').

##### Round Robin Group

Priority 8 is implemented as a Round Robin group that can be used for max. 8 master agents.

The Round Robin Group algorithm arbitrates among all master agents with Priority 8 that are requesting for bus access. The winner of the RR arbitration will be treated as a normal master request with priority 8 within the Priority Driven arbitration algorithm.

If only one master is mapped to a round robin group priority, the master's request will be treated as normal master request with priority 8 within the Priority driven arbitration algorithm.

After the winner of a round robin group is granted, the requesting master configured to priority 8 with the next higher index number will be selected as next winner of the round robin group. If there is no requesting master with a higher index number in the round robin group, the algorithm will start with the requesting master with the lowest index number in the round robin group, going to the next higher index number and so on. The index numbers are defined in the registers PRIOH, PRIOL.

##### Default Master Priorities after Reset

The default priority settings should be optimal for most applications. Where required, the arbitration setting can be adapted to the specific application needs (see 'Changing Master Priorities').

The default master agent priorities are described in the Appendix document, chapter BCU.

## Changing Master Priorities

Master priorities must be configured during ramp up as long as only one FPI master agent is active (e.g. CPU0 on SPB, SFI\_S2F on BBB). Master priorities must not be changed while multiple FPI master agents are active / enabled.

## Starvation Prevention

Starvation prevention takes care that even requesting low priority master agents will be granted after a period, where the period length can be controlled by BCU control registers. Because of the priority driven arbitration algorithm, it is possible that a lower-priority bus requestor may never be granted the bus if one or more higher-priority bus requestor continuously requests for, and receives, bus ownership. To protect against bus starvation of lower-priority masters, the starvation prevention mechanism of the BCU will detect such cases and momentarily raise the priority of the lower-priority requestor to the highest priority (above all other priorities), thereby guaranteeing it access.

Starvation protection employs a counter that is decremented each time an arbitration is performed on the connected FPI bus. When the counter is counted down to zero it is always re-loaded with the starvation period value in the BCU\_CON.SPC bit field. When the counter is counted down to zero, for each active bus request a request flag is stored in the BCU. This flag is cleared automatically when a master is granted the bus.

When the next counter period is finished and not all request flags were cleared, a starvation event happened. This masters related to the remaining request flags will now be set to the highest priority and will be granted in the order of their configured priority ranking.

If a master that is processing its transaction under starvation condition is retried, its corresponding request flag is automatically set again.

Starvation protection is permanently enabled. The sample period of the counter is programmed through bit field BCU\_CON.SPC. SPC must be larger than the number of masters. Its reset value is FF<sub>H</sub>.

### 4.10.3.2 FPI Bus Error Handling

When an error occurs on an FPI Bus, its BCU captures and stores data about the erroneous condition and generates a service request if enabled to do so. The error conditions that force an error-capture are:

- Error Acknowledge: An FPI Bus slave responds with an error to a transaction.
- Un-implemented Address: No FPI Bus slave responds to a transaction request.
- Time-out: A slave does not respond to a transaction request within a certain time window. The number of bus clock cycles that can elapse until a bus time-out is generated is defined by bit field BCU\_CON.TOUT.

When a transaction causes an error, the address and data phase signals of the transaction causing the error are captured and stored in registers.

- The Error Address Capture Register (BCU\_EADD) stores the 32-bit FPI Bus address that has been captured during the erroneous FPI Bus transaction.
- The Error Data Capture Registers (BCU\_EDAT) stores the 32-bit FPI Bus data bus information that has been captured during the erroneous FPI Bus transaction.
- The Error Control Capture Register (BCU\_ECON) stores status information of the bus error event.

If more than one FPI Bus transaction generates a bus error, only the first bus error is captured. After a bus error has been captured, the capture mechanism must be released again by software. The lock is removed by reading the register BCU\_ECON which clears the BCU\_ECON.ERRCNT bit field.

**Note:** It is recommended to read in a debug session register ECON last as this removes the lock and a new error can already modify the content of the other two registers EDAT and EADD.

If a write transaction from TriCore causes an error on the SPB, the originating master is not informed about this error as it is not an SPB master agent. With each bus error- capture event, a service request is generated, and an interrupt can be generated if enabled and configured in the corresponding service request register.

### Interpreting the BCU Control Register Error Information

Although the address and data values captured in registers BCU\_EADD and BCU\_EDAT, respectively, are self-explanatory, the captured FPI Bus control information needs some more explanation.

Register BCU\_ECON captures the state of the read (RDN), write (WRN), Supervisor Mode (SVM), acknowledge (ACK), ready (RDY), abort (ABT), time-out (TOUT), bus master identification lines (TAG) and transaction operation code (OPC) lines of the FPI Bus.

The SVM signal is set to 1 for an access in Supervisor Mode and set to 0 for an access in User Mode. The time-out signal indicates if there was no response on the bus to an access, and the programmed time (via BCU\_TOUT) has elapsed. TOUT is set to 1 in this case. An acknowledge code has to be driven by the selected slave during each data cycle of an access. These codes are listed in [Table 74](#).

**Table 74 FPI Bus Acknowledge Codes**

Code (ACK)	Description
00 <sub>B</sub>	NSC: No Special Condition.
01 <sub>B</sub>	Reserved
10 <sub>B</sub>	RTY: Retry. Slave can currently not respond to the access. Master needs to repeat the access later.
11 <sub>B</sub>	ERR: Bus Error, last data cycle is aborted.

Transactions on the FPI Bus are classified via a 4-bit operation code (see [Table 75](#)). Note that split transactions (OPC = 1000<sub>B</sub> to 1110<sub>B</sub>) are not used in this AURIX™ TC3xx Platform.

**Table 75 FPI Bus Operation Codes (OPC)**

OPC	Description
0000 <sub>B</sub>	Single Byte Transfer (8-bit)
0001 <sub>B</sub>	Single Half-Word Transfer (16-bit)
0010 <sub>B</sub>	Single Word Transfer (32-bit)
0100 <sub>B</sub>	2-Word Block Transfer
0101 <sub>B</sub>	4-Word Block Transfer
0110 <sub>B</sub>	8-Word Block Transfer
1111	No operation
0011 <sub>B</sub> , 0111 <sub>B</sub> , 1000 <sub>B</sub> - 1110 <sub>B</sub>	Reserved

### 4.10.4 FPI Bus Integrity Support

All transactions through the FPI Bus are protected with an Error Detection mechanism (EDC). The mechanism is defined and implemented in order to detect errors in between active FPI master and slave agents during a transaction e.g.:

- Transaction Address Phase errors
- Transaction Data Phase errors

- Errors in the transmitted Data
- Situations where no Slave, the wrong Slave or multiple Slaves respond to a transaction
- Situations where multiple Master are initiating transactions in parallel
- Active Master: Master that initiates a transaction
- Active Slave: Slave that is selected in the Address Phase

On detection of an error the Bus Control Unit (BCU) signals an alarm to the Safety Management Unit (SMU). For further analysis and decisions the BCU provides detailed informations where the error was detected.

The implemented solution ensures that only relevant errors will generate an alarm.

This includes

- The bus signals from the active master through the interconnect
- The bus signals from the active slave through the interconnect
- General control signals during a transaction to detect erroneous active slave / master interfaces
- FPI\_RDY='1' is used as qualifier for valid Data Phase / Address Phase / ECC informations

**Note:** *The integrity support provides in this implementation error detection capabilities only. For this, Error Correction Codes (ECC) codes will be used. The additional capabilities of the used ECC will be used for an improved error detection. Therefore the extension '\_ECC' is used in this chapter for related registers/FPI signals and SMU alarm signals.*

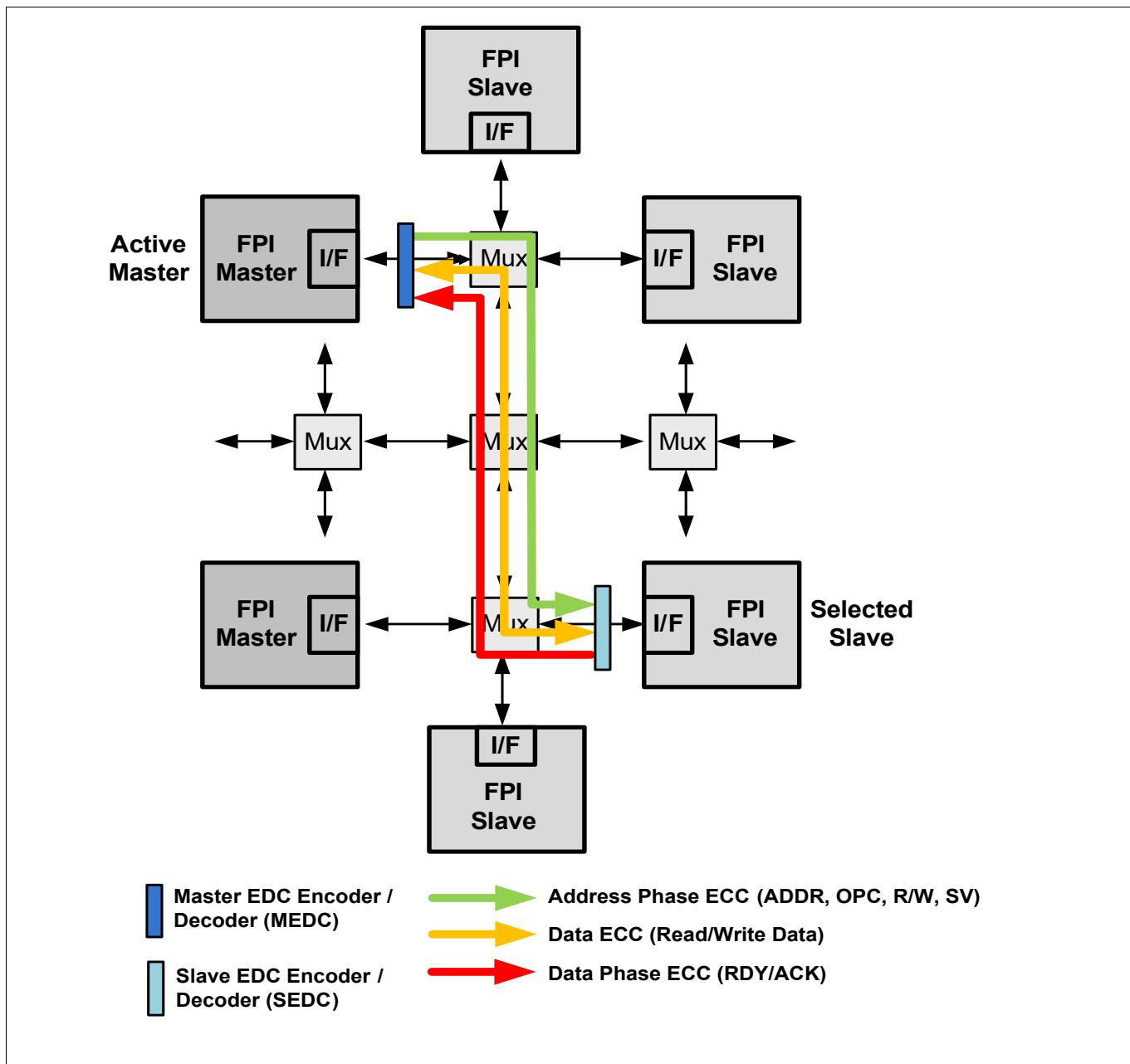


Figure 38 FPI EDC: Protection of Master → Slave data path through bus fabric

#### 4.10.4.1 Safety Support

##### BCU Alarm signals

Each instance of the FPI interconnect provides one alarm signal to the SMU, covering three alarm events:

- Alarm on detection of an FPI Bus Error by the BCU (Bus Error is a fatal error)
- Alarm on detection of a Timeout situation
- Alarm on detection of a FPI EDC error

For further analysis of the alarm:

- BCU captures informations of the first transaction that was terminated with Error Acknowledge or Timeout. The capture mechanism covers all relevant transaction informations inc. grant signals, 32-bit address, op-

code, TAG ID, 32-bit data, acknowledge code and FPI protocol control signals of the transactions address/data phase.

- BCU provides additional informations to identify the connected FPI slave / master interface where the FPI EDC error was detected

### **Capture of transaction informations on bus error**

The BCU capture mechanism capture the first transaction with a fatal error.

SW can release the mechanism again.

The capture mechanism covers all relevant transaction informations including:

- master grant signal status
- 32-bit address
- OP-code
- TAG ID
- 32-bit data
- acknowledge code
- FPI protocol control signals of the transactions address/data phase

The BCU can be configured to generate an interrupt for fatal errors (Error Acknowledge, Timeout, no slave responding) in parallel to the SMU alarm.

The related Service Request Nodes (SRN) and its Service Request Control registers (SRC) in the Interrupt Router (IR) module are:

- SRC\_BCUSPB (Service Request Control register related to the SBCU interrupt)
- SRC\_BCUBBB (Service Request Control register related to the EBCU interrupt)

### **Starvation Prevention**

The FPI arbitration algorithm in the BCU provides a starvation prevention mechanism. This feature of the BCU ensures that even requesting low priority master agents will be granted after a period, where the period length is defined via SBCU control registers.

The Starvation prevention is permanently enabled.

### **Time-out detection**

The BCU provides a time-out detection mechanism. This mechanism detects if a slave does not respond to a transaction request within a configurable time window. The time-out mechanism is also implemented in the FPI bus protocol. A dedicated TIMEOUT signal forces the active master / slave interfaces to release the bus if the BCU has detected a timeout situation. On detection of time-out situation the BCU signals an alarm to the SMU.

### **Register Access protection**

The BCU control registers are protected by the Register Access Protection, including the TAG ID based access protection mechanism (see ACCENO).

On detection of an Register Access Protection violation the BCU finish the transaction with Error Acknowledge. This will be detected by the BCU monitor function and signalled as alarm to the SMU.

### **Default Slave mechanism**

The FPI bus fabric implementation shows per default:

- FPI\_RDY = '1'

- FPI\_ACK = '11' (Error Ack)
- FPI\_DP\_ECC = '11' (Invalid ECC for RDY/ACK default)

This means that an EDC error will be detected for the active master interface when:

- the master access a reserved address (BCU sends alarm to SMU)
- the addressed FPI slave does not respond (BCU sends alarm to SMU)
- the default master is not driving the data phase signals during the default master address phase while the granted master looks at FPI\_RDY to identify the start/end of its address phase and the selected slave looks at FPI\_RDY to identify the start of its data phase (alarm from active master interface and active slave interface to BCU, BCU sends the alarm to SMU)

#### 4.10.4.2 FPI EDC Overview

The FPI EDC covers the FPI address, data and control signals from a FPI master agent output to the addressed FPI slave agent input during a transaction.

The FPI Bus Fabric includes

- a dedicated FPI Slave Encoder / Decoder Module (SEDM) for each connected Slave Interface
- a dedicated FPI Master Encoder / Decoder Module (MEDM) for each connected Master Interface

An Encoder / Decoder Module includes all ECC en-/decoder that are required to generate/check the ECC for the address and data phases of a transaction. The encoder/decoder module uses the FPI output enable signals of the related FPI master/slave interface to enable the encoding/decoding.

*Note: The FPI EDC is not working in the first 4 FPI bus cycles after an reset of the FPI bus fabric.*

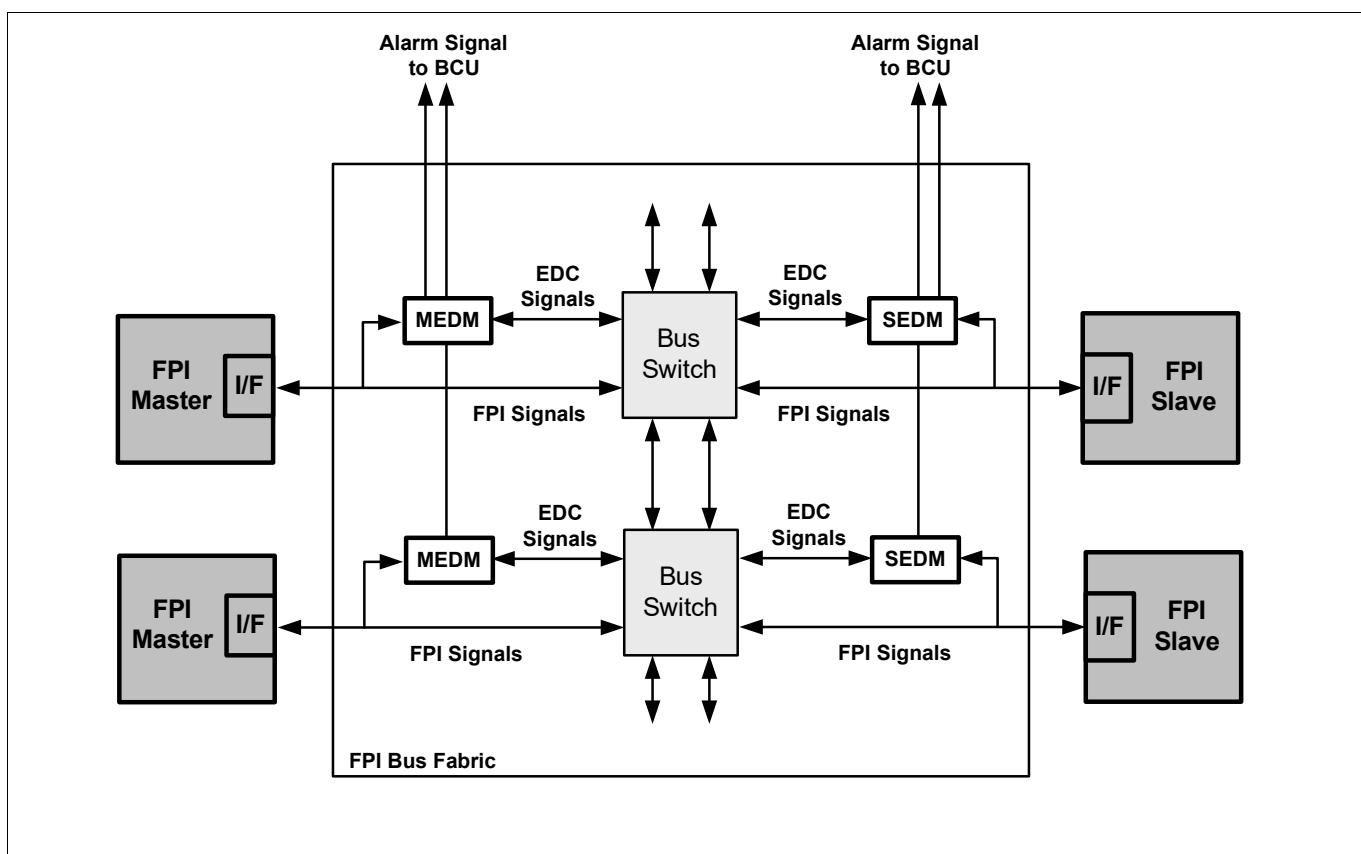


Figure 39 FPI EDCAURIX™ TC3xx Platform: encoder / decoder alarm signals

#### 4.10.4.3 Error Injection

The FPI EDC implementation allows to test all MEDM and SEDM decoder and the related alarm signal paths to the SMU by Error Injection.

The error injection is controlled via the BCU Error Generation control register BCU\_FEGEN.

- SBCU: Errors to test Slave related decoders are injected via the CPU0 MEDM encoder
- EBCU: Errors to test Slave related decoders are injected via the SFI\_S2F MEDM encoder
- Errors to test Slave related decoders are injected via the BCU SEDM encoder
- Errors to test other control signals (e.g. address/data-phase/data enable signals) are injected via the BCU.

##### MEDM decoder test

To test MEDM decoders with its related alarm signal path to the SMU the following sequence can be used:

- Enable the error injection related to the Encoder where an error has to be injected in BCU SEDM via the register BCU\_FEGEN (e.g. read data error).
- The FPI master related to the MEDM where the decoder alarm signal path has to be tested access the BCU with the an appropriate transaction (e.g. DMA reads a BCU control register in order to get an DMA related FPI EDC alarm)
- Disable the error injection by clearing the BCU\_FEGEN bits.

##### SEDM decoder test

To test SEDM decoders with its related alarm signal path to the SMU the following sequence can be used (example for an SPB test, BBB can be tested accordingly but with CPU access through the SFI\_S2F bridge):

- Enable the error injection related to the CPU0 Encoder where an error has to be injected in BCU MEDM via the register BCU\_FEGEN (e.g. address phase error).
- CPU0 has to access now the FPI slave related to the SEDM where the decoder alarm signal path has to be tested (e.g. CPU0 data access to FCE to get an alarm from the FCE related alarm signal on the SPB)
- Disable the error injection by clearing the BCU\_FEGEN bits.

**Note:** *The error injection mechanism is static. Means: errors will be injected in the related CPU0 MEDM encoder and/or the BCU related SEDM encoder as long the BCU\_FEGEN bit is set.*

#### 4.10.4.4 SPB: Mapping of ALARM signals to SBCU\_ALSTATx and SBCU\_ALCLRx registers

The mapping of the FPI EDC alarm signals (MEDM / SEDM modules) and the alarm from the SBCU internal output enable checks to the Alarm Status (SBCU\_ALSTATx; x=0-3) and Alarm Clear (SBCU\_ALCLRx; x=0-3) registers are described in the following tables.

**Note:** *The module interfaces mentioned in the tables below are per default of type 'Slave Agent'. Master agent interfaces are described as 'Master'.*

**Attention:** *These tables are showing the family mapping of alarm signals. For the effective mapping see the register specifications in Chapter 4.10.2.*

**Table 76 SBCU\_ALSTAT0, SBCU\_ALCLR0**

<b>BIT</b>	<b>Module</b>	<b>BIT</b>	<b>Module</b>
0	SBCU	16	QSPI2
1	DMA	17	QSPI3
2	Interrupt Router (IR)	18	QSPI4
3	SFI_F2S	19	QSPI5
4	SCU	20	FCE0
5	SMU	21	GETH1 (Slave)
6	PMC / SCR	22	STM0
7	MTU	23	STM1
8	IOM	24	STM2
9	Reserved	25	STM3
10	ASCLIN0 / ASCLIN1	26	STM4
11	ASCLIN2 / ASCLIN3	27	STM5
12	ASCLIN4 / ASCLIN5	28	PSI5
13	ASCLIN6 / ASCLIN7	29	PSI5S
14	QSPI0	30	ERAY0
15	QSPI1	31	ERAY1

**Table 77 SBCU\_ALSTAT1, SBCU\_ALCLR1**

<b>BIT</b>	<b>Module</b>	<b>BIT</b>	<b>Module</b>
0	GPT12	16	I2C0
1	CCU6	17	I2C1
2	GTM	18	HSSL1 (Slave)
3	MSC0	19	CONVCTRL (related to EVADC)
4	MSC1	20	ASCLIN8 / ASCLIN9
5	MSC2	21	ASCLIN10 / ASCLIN11
6	MSC3	22	ASCLIN12 / ASCLIN13
7	SENT	23	ASCLIN14 / ASCLIN15
8	GETH (Slave)	24	ASCLIN16 / ASCLIN17
9	EVADC	25	ASCLIN18 / ASCLIN19
10	EDSADC	26	ASCLIN20 / ASCLIN21
11	HSM	27	ASCLIN22 / ASCLIN23
12	HSSL0 (Slave)	28	HSPDM_SRAM_S (Slave)
13	CAN0	29	HSPDM_SFR_S (Slave)
14	CAN1	30	SDMMC (Slave)
15	CAN2	31	Cerberus (Slave)

**Table 78 SBCU\_ALSTAT2, SBCU\_ALCLR2**

<b>BIT</b>	<b>Module</b>	<b>BIT</b>	<b>Module</b>
0	P00	16	P25
1	P01	17	P26
2	P02	18	Reserved
3	Reserved	19	P30
4	P10	20	P31
5	P11	21	P32
6	P12	22	P33
7	P13	23	P34
8	P14	24	Reserved
9	P15	25	P40
10	Reserved	26	P41
11	P20	27	P50
12	P21	28	P51
13	P22	29	HSCT0_S
14	P23	30	HSCT1_S
15	P24	31	SBCU (Reset Driver)

**Table 79 SBCU\_ALSTAT3, SBCU\_ALCLR3**

<b>BIT</b>	<b>Output Enable Check</b>	<b>BIT</b>	<b>Module</b>
0	Address Phase (A_EN_N, Master)	16	DMA / Cerberus (Master)
1	Data Phase (ABORT_EN_N, Master)	17	Reserved
2	Data Phase (ACK_EN_N, Default Master and Slave)	18	SDMMC
3	Data Enables (D_EN_N, all Masters and Slaves)	19	HSSL0 (Master)
4	Reserved	20	HSSL1 (Master)
5	Reserved	21	Reserved
6	Reserved	22	CPU0 (Master)
7	Reserved	23	CPU1 (Master)
8	Reserved	24	CPU2 (Master)
9	Reserved	25	CPU3 (Master)
10	Reserved	26	CPU4 (Master)
11	Reserved	27	CPU5 (Master)
12	Reserved	28	HSM Register Master Interface (Master)
13	Reserved	29	HSM Cache Master Interface (Master)
14	Reserved	30	Reserved
15	Reserved	31	Reserved

#### 4.10.4.5 BBB: Mapping of ALARM signals to EBCU\_ALSTATx and EBCU\_ALCLRx registers

The mapping of the FPI EDC alarm signals (MEDM / SEDM modules) and the alarm from the EBCU internal output enable checks to the Alarm Status (EBCU\_ALSTATx; x=0-3) and Alarm Clear (EBCU\_ALCLRx; x=0-3) registers are described in the following tables.

**Note:** *The module interfaces mentioned in the tables below are per default of type 'Slave Agent'. Master agent interfaces are described as 'Master'.*

**Attention:** *These tables are showing the family / umbrella mapping of alarm signals. For the effective mapping see the register specifications in Chapter 4.10.4.*

**Table 80 EBCU\_ALSTAT0, EBCU\_ALCLR0**

BIT	Module	BIT	Module
0	EBCU	16	RIFO
1	MCDS	17	RIF1
2	AGBT	18	SPU0_S
3	Reserved	19	SPUCFG0_S
4	Reserved	20	SPU1_S
5	Reserved	21	SPUCFG1_S
6	EMEM XTMRAM	22	SPU Lockstep SFR
7	EMEM Control Register	23	CIF_S
8	EMEM0	24	Reserved
9	EMEM1	25	Reserved
10	EMEM2	26	Reserved
11	EMEM3	27	Reserved
12	Reserved	28	Reserved
13	Reserved	29	Reserved
14	Reserved	30	Reserved
15	Reserved	31	Reserved

**Table 81 EBCU\_ALSTAT1, EBCU\_ALCLR1**

BIT	Module	BIT	Module
0	Reserved	16	Reserved
1	Reserved	17	Reserved
2	Reserved	18	Reserved
3	Reserved	19	Reserved
4	Reserved	20	Reserved
5	Reserved	21	Reserved
6	Reserved	22	Reserved
7	Reserved	23	Reserved
8	Reserved	24	Reserved

**Table 81 EBCU\_ALSTAT1, EBCU\_ALCLR1 (cont'd)**

<b>BIT</b>	<b>Module</b>		<b>BIT</b>	<b>Module</b>
9	Reserved		25	Reserved
10	Reserved		26	Reserved
11	Reserved		27	Reserved
12	Reserved		28	Reserved
13	Reserved		29	Reserved
14	Reserved		30	Reserved
15	Reserved		31	Reserved

**Table 82 EBCU\_ALSTAT2, EBCU\_ALCLR2**

<b>BIT</b>	<b>Module</b>		<b>BIT</b>	<b>Module</b>
0	Reserved		16	Reserved
1	Reserved		17	Reserved
2	Reserved		18	Reserved
3	Reserved		19	Reserved
4	Reserved		20	Reserved
5	Reserved		21	Reserved
6	Reserved		22	Reserved
7	Reserved		23	Reserved
8	Reserved		24	Reserved
9	Reserved		25	Reserved
10	Reserved		26	Reserved
11	Reserved		27	Reserved
12	Reserved		28	Reserved
13	Reserved		29	Reserved
14	Reserved		30	Reserved
15	Reserved		31	EBCU (Reset Driver)

**Table 83 EBCU\_ALSTAT3, EBCU\_ALCLR3**

<b>BIT</b>	<b>Output Enable Check</b>		<b>BIT</b>	<b>Module</b>
0	Address Phase (A_EN_N, Master)		16	IOC32P
1	Data Phase (ABORT_EN_N, Master)		17	Reserved
2	Data Phase (ACK_EN_N, Default Master and Slave)		18	Reserved
3	Data Enables (D_EN_N, all Masters and Slaves)		19	IOC32E
4	Reserved		20	CIF
5	Reserved		21	Reserved

**Table 83 EBCU\_ALSTAT3, EBCU\_ALCLR3 (cont'd)**

<b>BIT</b>	<b>Output Enable Check</b>		<b>BIT</b>	<b>Module</b>
6	Reserved		22	SFI_S2F (Master)
7	Reserved		23	Reserved
8	Reserved		24	Reserved
9	Reserved		25	Reserved
10	Reserved		26	Reserved
11	Reserved		27	Reserved
12	Reserved		28	Reserved
13	Reserved		29	Reserved
14	Reserved		30	Reserved
15	Reserved		31	Reserved

#### 4.10.5 Debug

For debugging purposes, the BCU has the capability for breakpoint generation support. This OCDS debug capability is controlled by the Cerberus module and must be enabled by it (indicated by bit BCU\_DBCNTL.EO).

When BCU debug support has been enabled (EO = 1), any breakpoint request generated by the BCU to the Cerberus disarms the BCU breakpoint logic for further breakpoint requests. In order to rearm the BCU breakpoint logic again for the next breakpoint request generation, bit BCU\_DBCNTL.RA must be set. The status of the BCU breakpoint logic (armed or disarmed) is indicated by bit BCU\_DBCNTL.OA.

There are three types of trigger events:

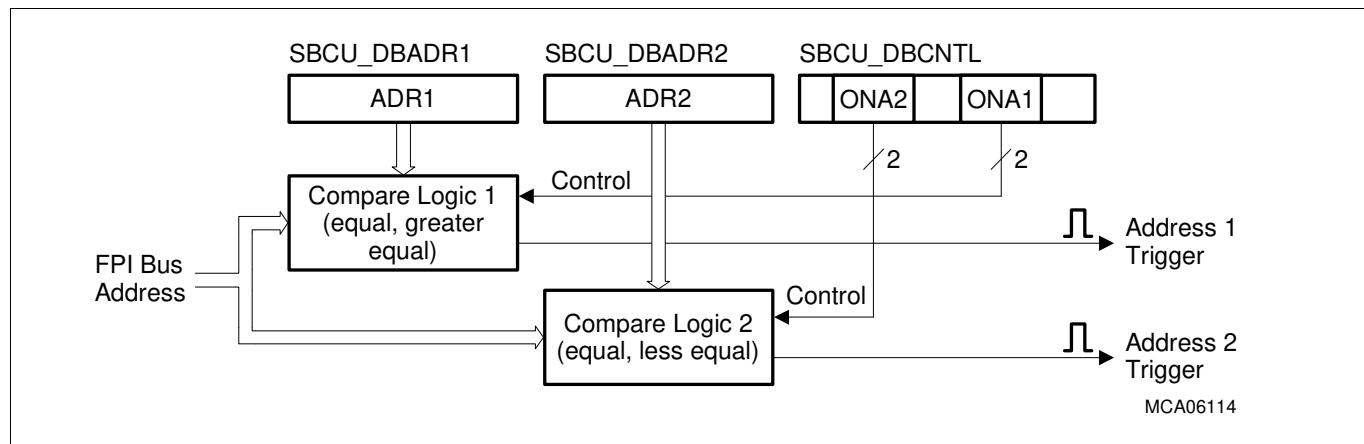
- Address triggers
- Signal triggers
- Grant triggers

##### 4.10.5.1 Address Trigger

The address debug trigger event conditions are defined by the contents of the BCU\_DBADR1, BCU\_DBADR2, and BCU\_DBCNTL registers. A wide range of possibilities arise for the creation of debug trigger events based on addresses. The following debug trigger events can be selected:

- Match on one signal address
- Match on one of two signal addresses
- Match on one address area
- Mismatch on one address area

Each pair of DBADRx registers and DBCNTL.ONAx bits determine one possible debug trigger event. The combination of these two possible debug trigger events defined by DBCNTL.CONCOM1 determine the address debug trigger event condition.



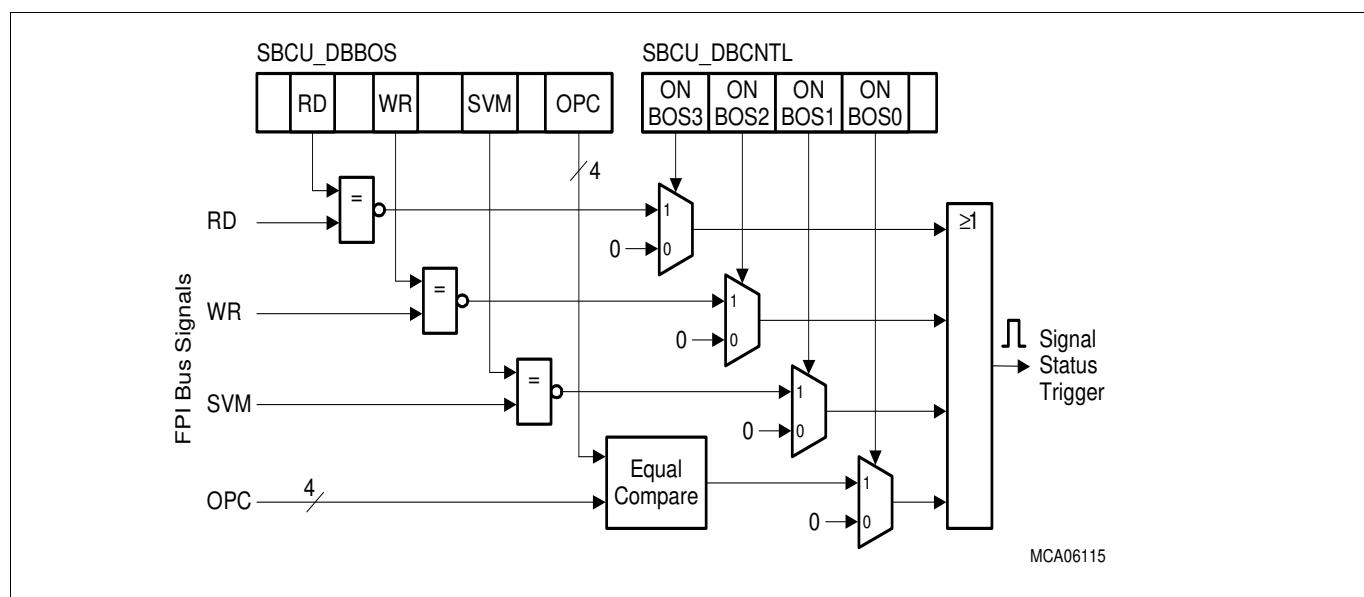
**Figure 40 Address Trigger Generation**

#### **4.10.5.2 Signal Status Trigger**

The signal status debug trigger event conditions are defined by the contents of the BCU\_DBBOS and BCU\_DBCNTL registers. Depending on the selected configuration a wide range of possibilities arise for the creation of a debug trigger event based on FPI Bus status signals. Possible combinations are:

- Match on a single signal status
  - Match on a multiple signal status

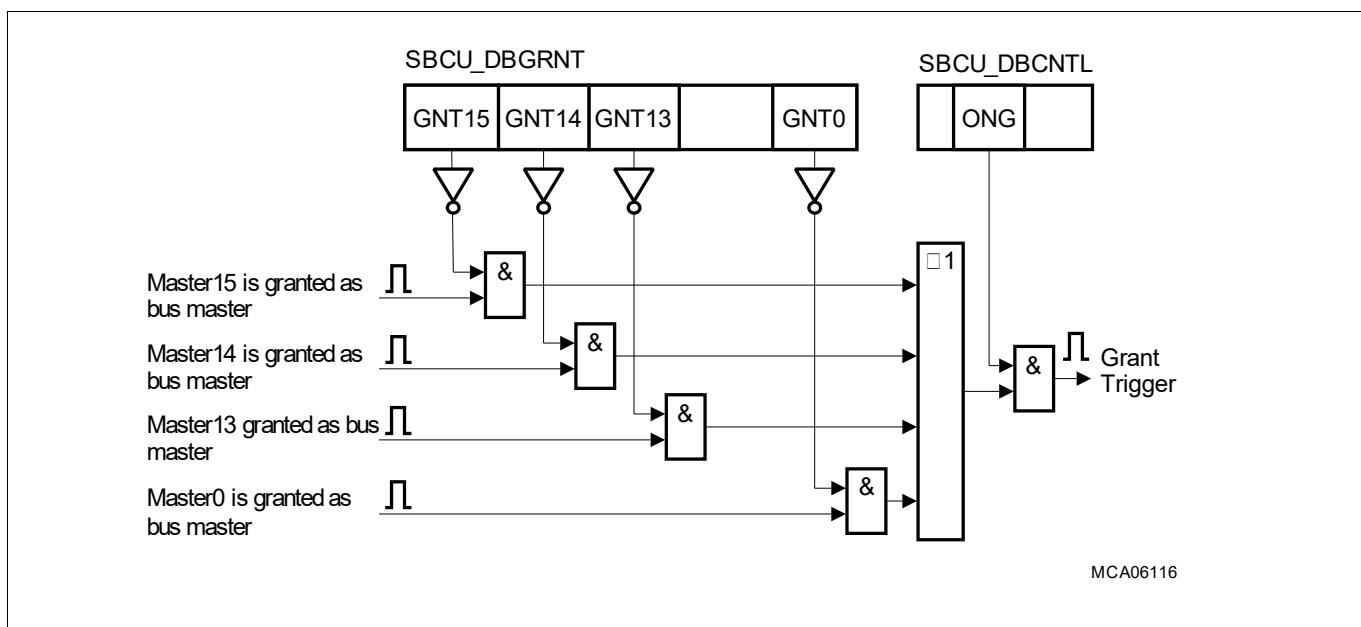
With the multiple signal match conditions, all single signal match conditions are combined with a logical **AND** to the signal status debug trigger event signal. The selection whether or not a single match condition is selected can be enabled/disabled selectively for each condition via the BCU\_DBCNTL.ONBOSx bits.



**Figure 41 Signal Status Trigger Generation**

#### 4.10.5.3 Grant Trigger

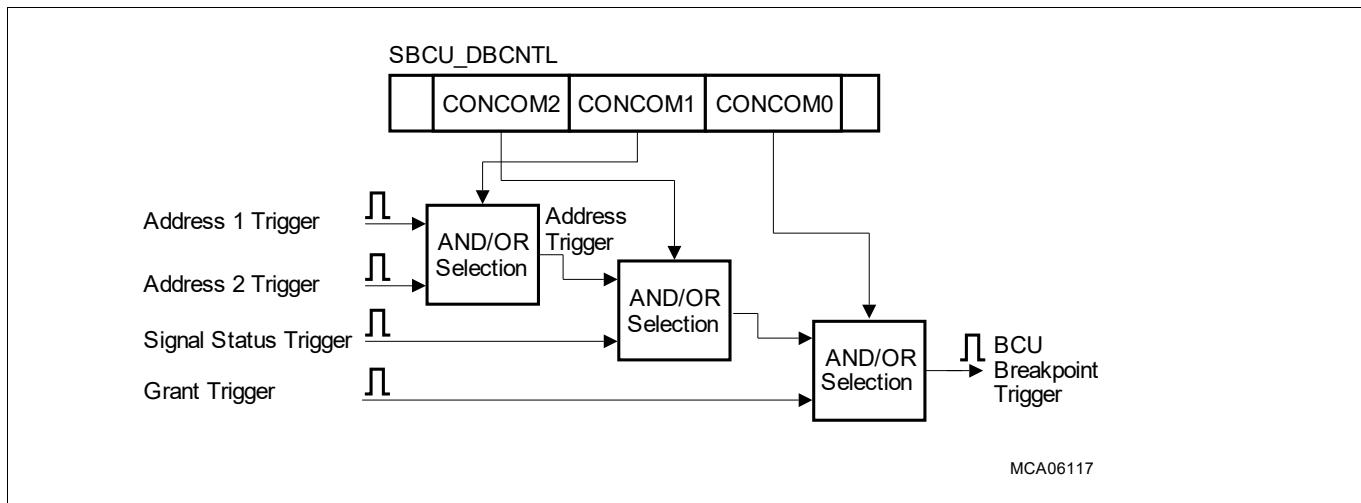
The signal status debug trigger event conditions are defined via the registers SBCU\_DBGRNT and SBCU\_DBCNTL. Depending on the configuration of these registers, any combination of FPI Bus master trigger events can be configured. Only the enabled masters in the SBCU\_DBGRNT register are of interest for the grant debug trigger event condition. The grant debug trigger event condition can be enabled/disabled via bit SBCU\_DBCNTL.ONG (see [Figure 42](#)).



**Figure 42** Grant Trigger Generation

#### 4.10.5.4 Combination of Trigger Events

The combination of the four debug trigger signals to the single BCU breakpoint trigger event is defined via the bits CONCOM[2:0] of register BCU\_DBCNTL (see [Figure 43](#)). The two address triggers are combined to one address trigger that is further combined with signal status and grant trigger signals. A logical AND or OR combination can be selected for the BCU breakpoint trigger generation.



**Figure 43 BCU Breakpoint Trigger Combination Logic**

#### 4.10.5.5 BCU Breakpoint Generation Examples

This section gives three examples of how BCU debug trigger events are programmed.

##### OCDS Debug Example 1

- Task: Generation of a BCU debug trigger event on any SPB write access to address  $00002004_H$  or  $000020A0_H$  by an SPB master.

For this task, the following programming settings for the BCU breakpoint logic must be executed:

- Writing BCU\_DBADR1 =  $0000\ 2004_H$
- Writing BCU\_DBADR2 =  $0000\ 20A0_H$
- Writing BCU\_DBCNTL =  $C1115010_H$ :
  - $ONBOS[3:0] = 1100_B$  means that no signal status trigger is generated (disabled) for a read signal match AND write signal match condition according to the settings of bits RD and WR in register BCU\_DBBOS. Debug trigger event generation for Supervisor Mode signal match and op-code signal match condition is disabled.
  - $ONA2 = 01_B$  means that the equal match condition for debug address 2 register is selected.
  - $ONA1 = 01_B$  means that the equal match condition for debug address 1 register is selected.
  - $ONG = 1$  means that the grant debug trigger is enabled.
  - $CONCOM[2:0] = 101_B$  means that the address trigger is created by address trigger 1 OR address trigger 2 ( $CONCOM1 = 0$ ), and that the grant trigger is ANDed with the address trigger ( $CONCOM0 = 1$ ), and that the signal status trigger is ANDed with the address trigger ( $CONCOM2 = 1$ ).
  - $RA = 1$  means that the BCU breakpoint logic is rearmed.
- Writing BCU\_DBGRNT =  $FFFFFD7_H$ : means that the grant trigger for the SPB master is enabled.

5. Writing  $\text{BCU\_DBBOS} = 00001000_{\text{H}}$ :  
means that the signal status trigger is generated on a write transfer and not on a read transfer.

### OCDS Debug Example 2

- Task: generation of a BCU debug trigger event on any half-word access in User Mode to address area  $01FFFFFF_{\text{H}}$  to  $02FFFFFF_{\text{H}}$  by any master.

For this task, the following programming settings for the BCU breakpoint logic must be executed:

- Writing  $\text{BCU\_DBADR1} = 01FFFFFF_{\text{H}}$
- Writing  $\text{BCU\_DBADR2} = 02FFFFFF_{\text{H}}$
- Writing  $\text{BCU\_DBCNTL} = 32206010_{\text{H}}$ :
  - $\text{ONBOS}[3:0] = 0011_B$  means that the signal status trigger is disabled for a read or for write signal status match but enabled for Supervisor Mode match AND op-code match conditions according to the settings of bit SVM and bit field OPC in register  $\text{BCU\_DBBOS}$ .
  - $\text{ONA2} = 10_B$  means that the address 2 trigger is generated if the FPI Bus address is less or equal to  $\text{BCU\_DBADR2}$ .
  - $\text{ONA1} = 10_B$  means that the address 1 trigger is generated if the FPI Bus address is greater or equal to  $\text{BCU\_DBADR1}$ .
  - $\text{ONG} = 0$  means that the grant debug trigger is disabled.
  - $\text{CONCOM}[2:0] = 110_B$  means that the address trigger is created by address trigger 1 AND address trigger 2 ( $\text{CONCOM1} = 1$ ), and that the grant trigger is OR-ed with the address trigger ( $\text{CONCOM0} = 0$ ), and that the signal status trigger is AND-ed with the address trigger ( $\text{CONCOM2} = 1$ ).
  - $\text{RA} = 1$  means that the BCU breakpoint logic is rearmed.
- Writing  $\text{BCU\_DBGRNT} = FFFFFFFF_{\text{H}}$ :  
means that no grant trigger for SPB masters is selected (“don’t care” because also disabled by  $\text{ONG} = 0$ ).
- Writing  $\text{BCU\_DBBOS} = 00000001_{\text{H}}$ :  
means that the signal status trigger is generated for read ( $\text{RD} = 0$ ) and write ( $\text{WR} = 0$ ) half-word transfers ( $\text{OPC} = 0001_B$ ) in User Mode ( $\text{SVM} = 0$ ).

#### 4.10.6 Registers

**Table 84 Register Overview - BCU (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
ID	Module Identification Register	0008 <sub>H</sub>	U,SV	BE	Application Reset	57
CON	BCU Control Register	0010 <sub>H</sub>	U,SV	SV,P	Application Reset	57
PRIOH	Arbiter Priority Register High	0014 <sub>H</sub>	U,SV	SV,E,P	Application Reset	58
PRIOL	Arbiter Priority Register Low	0018 <sub>H</sub>	U,SV	SV,E,P	Application Reset	59
ECON	BCU Error Control Capture Register	0020 <sub>H</sub>	U,SV	SV,P	Application Reset	59
EADD	BCU Error Address Capture Register	0024 <sub>H</sub>	U,SV	SV,P	Application Reset	61
EDAT	BCU Error Data Capture Register	0028 <sub>H</sub>	U,SV	SV,P	Application Reset	61
DBCNTL	BCU Debug Control Register	0030 <sub>H</sub>	U,SV	SV,P	Debug Reset	62
DBGRNT	BCU Debug Grant Mask Register	0034 <sub>H</sub>	U,SV	SV,P	Debug Reset	64
DBADR1	BCU Debug Address 1 Register	0038 <sub>H</sub>	U,SV	SV,P	Debug Reset	65
DBADR2	BCU Debug Address 2 Register	003C <sub>H</sub>	U,SV	SV,P	Debug Reset	65
DBBOS	BCU Debug Bus Operation Signals Register	0040 <sub>H</sub>	U,SV	SV,P	Debug Reset	66
DBGNTT	BCU Debug Trapped Master Register	0044 <sub>H</sub>	U,SV	BE	Debug Reset	67
DBADRT	BCU Debug Trapped Address Register	0048 <sub>H</sub>	U,SV	BE	Debug Reset	67
DBBOST	BCU Debug Trapped Bus Operation Signals Register	004C <sub>H</sub>	U,SV	BE	Debug Reset	68
DBDAT	BCU Debug Data Status Register	0050 <sub>H</sub>	U,SV	BE	Debug Reset	70
ALSTATx	BCU EDC Alarm Status Register x	0060 <sub>H</sub> +x *4	U,SV	SV,P	Application Reset	70
ALCLRx	BCU EDC Alarm Clear Register x	0070 <sub>H</sub> +x *4	U,SV	SV,P	Application Reset	70
ALCTRL	BCU EDC Alarm Control Register	0080 <sub>H</sub>	U,SV	SV,P	Application Reset	71
FEGEN	FPI Error Generation Control Register	0084 <sub>H</sub>	U,SV	SV,SE	Application Reset	72

Table 84 Register Overview - BCU (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
ACCEN1	Access Enable Register 1	00F8 <sub>H</sub>	U,SV	SV,SE	Application Reset	74
ACCENO	Access Enable Register 0	00FC <sub>H</sub>	U,SV	SV,SE	Application Reset	74

#### 4.10.6.1 Registers Description

##### Module Identification Register

The identification register allows the programmer version-tracking of the module. The table below shows the identification register which is implemented in the SBCU module.

ID															
Module Identification Register (0008 <sub>H</sub> ) Application Reset Value: 0000 6AXX <sub>H</sub>															
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16															
0															
r															
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0															
MOD_NUMBER MOD_REV															
r r															

Field	Bits	Type	Description
MOD_REV	7:0	r	<b>Module Revision Number</b> This bit field defines the module revision number. The value of a module revision starts with 01H (first revision).
MOD_NUMBE R	15:8	r	<b>Module Number Value</b> This bit field defines a module identification number. The value for the LBCU module is 006AH.
0	31:16	r	<b>Reserved</b> Read as 0; should be written with 0.

##### BCU Control Register

The SBCU Control Register controls the overall operation of the SBCU, including setting the starvation sample period, the bus time-out period, enabling starvation-protection mode, and error handling.

## CON

## BCU Control Register

(0010<sub>H</sub>)Application Reset Value: FF09 FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>SPC</b>															
															<b>DBG</b>
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>TOUT</b>															

Field	Bits	Type	Description
<b>TOUT</b>	15:0	rw	<b>BCU Bus Time-Out Value</b> The bit field determines the number of System Peripheral Bus time-out cycles. Default after reset is FFFF <sub>H</sub> (= 65536 bus cycles).  Note: TOUT value must be >= 5.
<b>DBG</b>	16	rw	<b>BCU Debug Trace Enable</b> The bit enables/disables the error capture mechanism for the registers BCU_ECON, BCU_EADD, BCU_EDAT. The bit does not affect the SMU alarm or the BCU interrupt that are send on detection case of an error condition. 0 <sub>B</sub> SBCU debug trace disabled 1 <sub>B</sub> SBCU debug trace enabled (default after reset)
<b>SPC</b>	31:24	rw	<b>Starvation Period Control</b> Determines the sample period for the starvation counter. Must be larger than the number of masters. The reset value is FF <sub>H</sub> .
<b>0</b>	18:17, 23:20	r	<b>Reserved</b> Read as 0; should be written with 0.
<b>1</b>	19	r	<b>Reserved</b> Read as 1; should be written with 0.

## Arbiter Priority Register High

## PRIOH

## Arbiter Priority Register High

(0014<sub>H</sub>)Application Reset Value: FEDC BA98<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>MASTER15</b>				<b>MASTER14</b>				<b>MASTER13</b>				<b>MASTER12</b>			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>MASTER11</b>				<b>MASTER10</b>				<b>MASTER9</b>				<b>MASTER8</b>			

Field	Bits	Type	Description
<b>MASTERi (i=8-15)</b>	4*i-29:4*i-32	rw	<b>Master i Priority</b> This bit field defines the priority on the SPB for master i access to the SPB. A lower number has a higher priority in the arbitration round than a higher one.

### Arbiter Priority Register Low

#### PRIOL

**Arbiter Priority Register Low** **(0018<sub>H</sub>)** **Application Reset Value: 7654 3210<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>MASTER7</b>				<b>MASTER6</b>				<b>MASTER5</b>				<b>MASTER4</b>			
rw					rw				rw				rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>MASTER3</b>				<b>MASTER2</b>				<b>MASTER1</b>				<b>MASTER0</b>			
rw					rw				rw				rw		

Field	Bits	Type	Description
<b>MASTERi (i=0-7)</b>	4*i+3:4*i	rw	<b>Master i Priority</b> This bit field defines the priority on the SPB for master i access to the SPB. A lower number has a higher priority in the arbitration round than a higher one.

### BCU Error Control Capture Register

#### ECON

**BCU Error Control Capture Register** **(0020<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>OPC</b>				<b>TAG</b>				<b>RDN</b>		<b>WRN</b>	<b>SVM</b>	<b>ACK</b>		<b>ABT</b>	
rw					rw				rw				rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>RDY</b>	<b>TOUT</b>	<b>ERRCNT</b>													
rwh	rwh														

Field	Bits	Type	Description
<b>ERRCNT</b>	13:0	rwh	<b>FPI Bus Error Counter</b> ERRCNT is incremented on every occurrence of an FPI Bus error. EERRCNT is reset to 0 after the ECON register is read. <sup>1)</sup>

Field	Bits	Type	Description
<b>TOUT</b>	14	rwh	<b>State of FPI Bus Time-Out Signal</b> This bit indicates the state of the time-out signal at an FBI Bus error. $0_B$ No time-out occurred $1_B$ Time-out has occurred
<b>RDY</b>	15	rwh	<b>State of FPI Bus Ready Signal</b> This bit indicates the state of the ready signal at an FBI Bus error. $0_B$ Wait state(s) have been inserted. Ready signal was active $1_B$ Ready signal was inactive
<b>ABT</b>	16	rwh	<b>State of FPI Bus Abort Signal</b> This bit indicates the state of the abort signal at an FBI Bus error. $0_B$ Master has aborted an FPI Bus transfer. Abort signal was active $1_B$ Abort signal was inactive
<b>ACK</b>	18:17	rwh	<b>State of FPI Bus Acknowledge Signals</b> This bit field indicates the acknowledge code that has been output by the selected slave at an FPI Bus error. Coding see <a href="#">Table 74</a> . $00_B$ NSC: No special case $01_B$ Reserved $10_B$ RTY: Retry $11_B$ ERR: Bus Error
<b>SVM</b>	19	rwh	<b>State of FPI Bus Supervisor Mode Signal</b> This bit indicates whether the FPI Bus error occurred in Supervisor Mode or in User Mode. $0_B$ Transfer was initiated in User Mode $1_B$ Transfer was initiated in Supervisor Mode
<b>WRN</b>	20	rwh	<b>State of FPI Bus Write Signal</b> This bit indicates whether the FPI Bus error occurred at a write cycle (see <a href="#">Table 85</a> ).
<b>RDN</b>	21	rwh	<b>State of FPI Bus Read Signal</b> This bit indicates whether the FPI Bus error occurred at a read cycle (see <a href="#">Table 85</a> ).
<b>TAG</b>	27:22	rwh	<b>FPI Bus Master Tag Number Signals</b> This bit field indicates the FPI Bus master TAG number (definitions see <a href="#">Table 86</a> ).
<b>OPC</b>	31:28	rwh	<b>FPI Bus Operation Code Signals</b> The FPI Bus operation codes are defined in <a href="#">Table 75</a> .

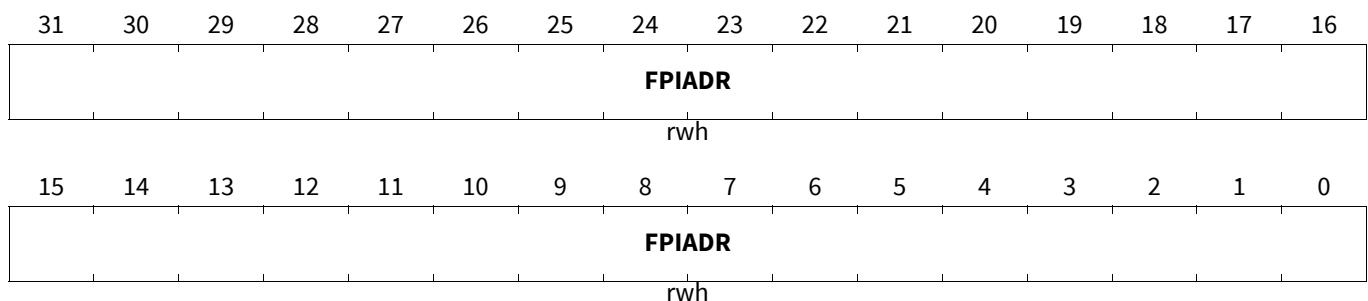
- 1) Aborted accesses to a 0 wait state SPB slave may also increment ERRCNT when the slave generates an error acknowledge.

**Table 85 FPI Bus Read/Write Error Indication**

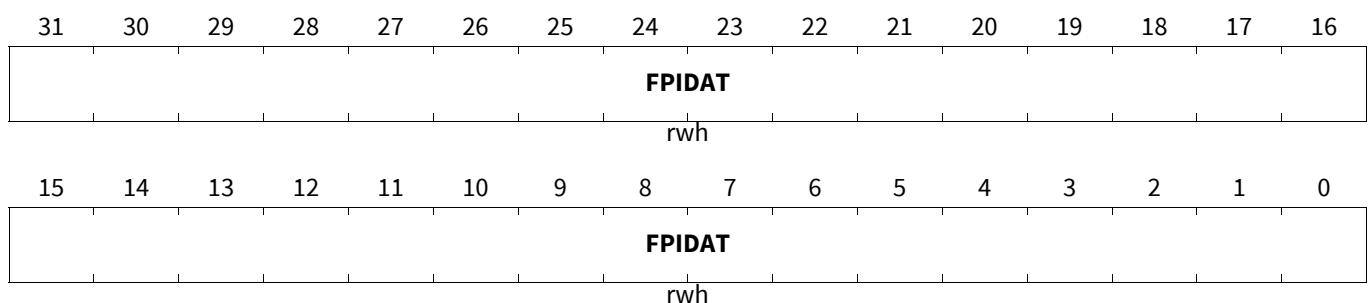
RD	WR	FPI Bus Cycle
0	0	FPI Bus error occurred at the read transfer of a read-modify-write transfer.
0	1	FPI Bus error occurred at a read cycle of a single transfer.

**Table 85 FPI Bus Read/Write Error Indication (cont'd)**

<b>RD</b>	<b>WR</b>	<b>FPI Bus Cycle</b>
1	0	FPI Bus error occurred at a write cycle of a single transfer or at the write cycle of a read-modify-write transfer.
1	1	Does not occur.

**BCU Error Address Capture Register****EADD****BCU Error Address Capture Register (0024<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>FPIADR</b>	31:0	rwh	<b>Captured FPI Bus Address</b> This bit field holds the 32-bit FPI Bus address that has been captured at an FPI Bus error. Note that if multiple bus errors occurred, only the address of the first bus error is captured.

**BCU Error Data Capture Register****EDAT****BCU Error Data Capture Register (0028<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>FPIDAT</b>	31:0	rwh	<b>Captured FPI Bus Data</b> This bit field holds the 32-bit FPI Bus data that has been captured at an FPI Bus error. Note that if multiple bus errors occurred, only the data of the first bus error is captured.

## BCU Debug Control Register

## DBCNTL

BCU Debug Control Register (0030<sub>H</sub>) Debug Reset Value: 0000 7003<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>ONBO S3</b>	<b>ONBO S2</b>	<b>ONBO S1</b>	<b>ONBO S0</b>	<b>0</b>	<b>ONA2</b>	<b>0</b>	<b>ONA1</b>								<b>ONG</b>
rw	rw	rw	rw	r	rw	r	rw	r	rw	rw	r	r	r	rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>0</b>	<b>CONC OM2</b>	<b>CONC OM1</b>	<b>CONC OMO</b>			<b>0</b>	<b>HSMD BGEN</b>	<b>HSMTRTREN</b>	<b>RA</b>		<b>0</b>	<b>OA</b>	<b>EO</b>		
r	rw	rw	rw	r		r	r	r	w	r	r	rh	r		

Field	Bits	Type	Description
<b>EO</b>	0	r	<b>Status of BCU Debug Support Enable</b> This bit is controlled by the Cerberus and enables the BCU debug support.  $0_B$ BCU debug support is disabled $1_B$ BCU debug support is enabled (default after reset)
<b>OA</b>	1	rh	<b>Status of BCU Breakpoint Logic</b> The OA bit is set by writing a 1 to bit RA. When OA is set, registers DBGNTT, DBADRT and DBDAT are reset. Also DBBOST is reset with the exception of the bit field FPIRST.  $0_B$ The BCU breakpoint logic is disarmed. Any further breakpoint activation is discarded $1_B$ The BCU breakpoint logic is armed
<b>RA</b>	4	w	<b>Rearm BCU Breakpoint Logic</b> Writing a 1 to this bit rearms BCU breakpoint logic and sets bit OA = 1. RA is always reads as 0.
<b>HSMTRTREN</b>	6:5	r	<b>Status of HSM Transaction Trace Logic</b> $00_B$ HSM transaction tracing and capturing of HSM transactions in the BCU is disabled $01_B$ HSM transaction tracing and capturing of HSM transactions in the BCU is disabled $10_B$ HSM transaction address can be traced, capturing of HSM transactions in the BCU is allowed $11_B$ HSM transaction address and data can be traced, capturing of HSM transactions in the BCU is allowed
<b>HSMDBGEN</b>	7	r	<b>Status of HSM Debug Mode</b> $0_B$ HSM module is not in debug mode $1_B$ HSM module is in debug mode

Field	Bits	Type	Description
<b>CONCOM0</b>	12	rw	<p><b>Grant and Address Trigger Relation</b></p> <p><b>0<sub>B</sub></b> The grant phase trigger condition and the address trigger condition (see CONCOM1) are combined with a logical OR for further control</p> <p><b>1<sub>B</sub></b> The grant phase trigger condition and the address trigger condition (see CONCOM1) are combined with a logical AND for further control.</p>
<b>CONCOM1</b>	13	rw	<p><b>Address 1 and Address 2 Trigger Relation</b></p> <p><b>0<sub>B</sub></b> Address 1 trigger condition and address 2 trigger condition are combined with a logical OR to the address trigger condition for further control</p> <p><b>1<sub>B</sub></b> Address 1 trigger condition and address 2 trigger condition are combined with a logical AND to the address trigger condition for further control</p>
<b>CONCOM2</b>	14	rw	<p><b>Address and Signal Trigger Relation</b></p> <p><b>0<sub>B</sub></b> Address trigger condition (see CONCOM1) and signal status trigger conditions are combined with a logical OR for further control</p> <p><b>1<sub>B</sub></b> Address phase trigger condition (see CONCOM1) and the signal status trigger conditions are combined with a logical AND for further control</p>
<b>ONG</b>	16	rw	<p><b>Grant Trigger Enable</b></p> <p><b>0<sub>B</sub></b> No grant debug event trigger is generated</p> <p><b>1<sub>B</sub></b> The grant debug event trigger is enabled and generated according the settings of register DBGRNT</p>
<b>ONA1</b>	21:20	rw	<p><b>Address 1 Trigger Control</b></p> <p><b>00<sub>B</sub></b> No address 1 trigger is generated</p> <p><b>01<sub>B</sub></b> An address 1 trigger event is generated if the FPI Bus address is equal to DBADR1</p> <p><b>10<sub>B</sub></b> An address 1 trigger event is generated if FPI Bus address is greater or equal to DBADR1</p> <p><b>11<sub>B</sub></b> same as 00<sub>B</sub></p>
<b>ONA2</b>	25:24	rw	<p><b>Address 2 Trigger Control</b></p> <p><b>00<sub>B</sub></b> No address 2 trigger is generated.</p> <p><b>01<sub>B</sub></b> An address 2 trigger event is generated if the FPI Bus address is equal to DBADR2</p> <p><b>10<sub>B</sub></b> An address 2 trigger event is generated if FPI Bus address is less or equal to DBADR2</p> <p><b>11<sub>B</sub></b> same as 00<sub>B</sub></p>

Field	Bits	Type	Description
<b>ONBOS0</b>	28	rw	<b>Op code Signal Status Trigger Condition</b> $0_B$ A signal status trigger is generated for all FPI Bus op-codes except a “no operation” op-code $1_B$ A signal status trigger is generated if the FPI Bus op-code matches the op-code as defined in DBBOS.OPC
<b>ONBOS1</b>	29	rw	<b>Supervisor Mode Signal Trigger Condition</b> $0_B$ The signal status trigger generation for the FPI Bus Supervisor Mode signal is disabled. $1_B$ A signal status trigger is generated if the FPI Bus Supervisor Mode signal state is equal to the value of DBBOS.SVM
<b>ONBOS2</b>	30	rw	<b>Write Signal Trigger Condition</b> $0_B$ The signal status trigger generation for the FPI Bus write signal is disabled. $1_B$ A signal status trigger is generated if the FPI Bus write signal state is equal to the value of DBBOS.WR
<b>ONBOS3</b>	31	rw	<b>Read Signal Trigger Condition</b> $0_B$ The signal status trigger generation for the FPI Bus read signal is disabled. $1_B$ A signal status trigger is generated if the FPI Bus read signal state is equal to the value of DBBOS.RD
<b>0</b>	3:2, 11:8, 15, 19:17, 23:22, 27:26	r	<b>Reserved</b> Read as 0; should be written with 0.

### BCU Debug Grant Mask Register

#### DBGRNT

#### BCU Debug Grant Mask Register

(0034<sub>H</sub>)Debug Reset Value: 0000 FFFF<sub>H</sub>

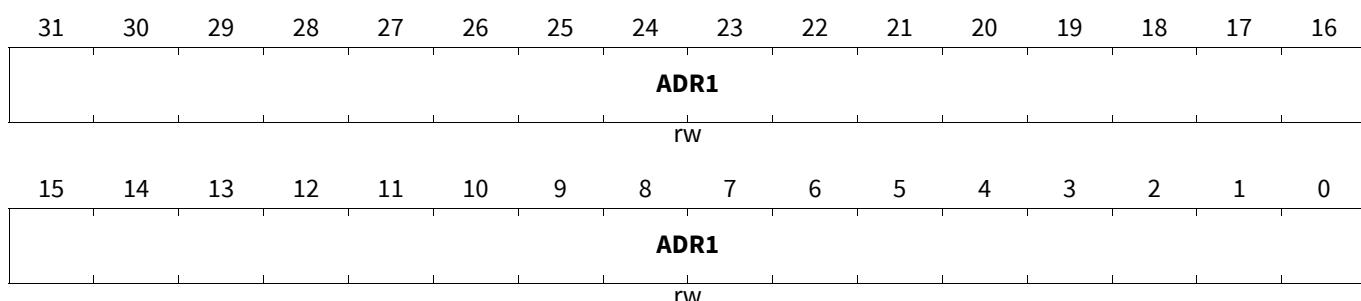
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FPIGN T15	FPIGN T14	FPIGN T13	FPIGN T12	FPIGN T11	FPIGN T10	FPIGN T9	FPIGN T8	FPIGN T7	FPIGN T6	FPIGN T5	FPIGN T4	FPIGN T3	FPIGN T2	FPIGN T1	FPIGN T0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FPIGNT <sub>i</sub> (i=0-15)	i	rw	<b>Master i Grant Trigger Enable</b> $0_B$ FPI Bus transactions with Master i as bus master are enabled for grant trigger event generation $1_B$ FPI Bus transactions with Master i as bus master are disabled for grant trigger event generation
0	31:16	r	<b>Reserved</b> Read as 0; should be written with 0.

### BCU Debug Address 1 Register

#### DBADR1

##### BCU Debug Address 1 Register

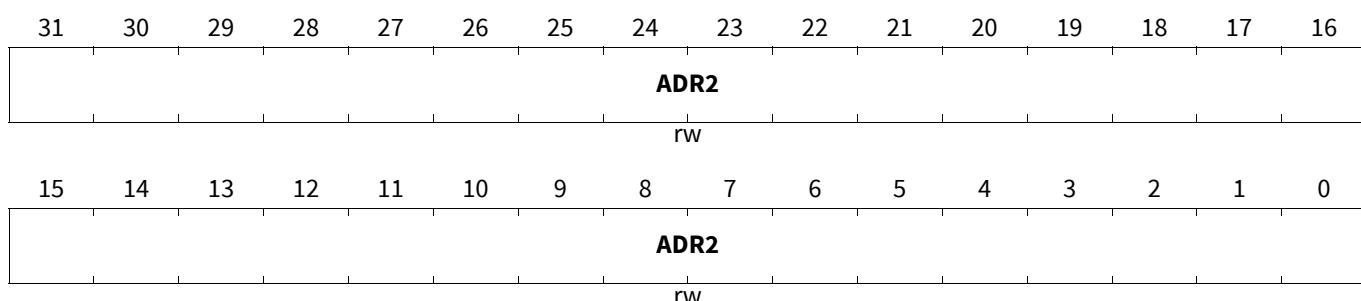
(0038<sub>H</sub>)Debug Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
ADR1	31:0	rw	<b>Debug Trigger Address 1</b> This register contains the address for the address 1 trigger event generation.

### BCU Debug Address 2 Register

#### DBADR2

##### BCU Debug Address 2 Register

(003C<sub>H</sub>)Debug Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
ADR2	31:0	rw	<b>Debug Trigger Address 2</b> This register contains the address for the address 2 trigger event generation.

## BCU Debug Bus Operation Signals Register

### DBBOS

**BCU Debug Bus Operation Signals Register (0040<sub>H</sub>)**      **Debug Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
r			<b>RD</b>	r		<b>0</b>		<b>WR</b>	r	<b>0</b>		<b>SVM</b>			<b>OPC</b>

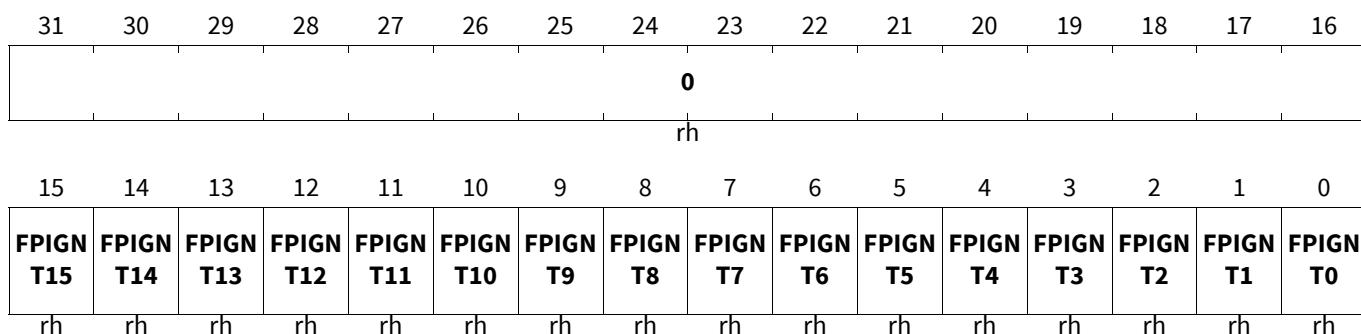
Field	Bits	Type	Description
<b>OPC</b>	3:0	rw	<b>Opcode for Signal Status Debug Trigger</b> This bit field determines the type (opcode) of an FPI Bus transaction for which a signal status debug trigger event is generated (if enabled by DBCNTL.ONBOS0 = 1). Other bit combinations are reserved. 0 <sub>H</sub> Trigger on single byte transfer selected 1 <sub>H</sub> Trigger on single half-word transfer selected 2 <sub>H</sub> Trigger on single word transfer selected 4 <sub>H</sub> Trigger on 2-word block transfer selected 5 <sub>H</sub> Trigger on 4-word block transfer selected 6 <sub>H</sub> Trigger on 8-word block transfer selected F <sub>H</sub> Trigger on no operation selected
<b>SVM</b>	4	rw	<b>SVM Signal for Status Debug Trigger</b> This bit determines the mode of an FPI Bus transaction for which a signal status debug trigger event is generated (if enabled by DBCNTL.ONBOS1 = 1). 0 <sub>B</sub> Trigger on User Mode selected 1 <sub>B</sub> Trigger on Supervisor Mode selected
<b>WR</b>	8	rw	<b>Write Signal for Status Debug Trigger</b> This bit determines the state of the WR signal of an FPI Bus transaction for which a signal status debug trigger event is generated (if enabled by DBCNTL.ONBOS2 = 1). 0 <sub>B</sub> Trigger on a single write transfer or write cycle of an atomic transfer selected 1 <sub>B</sub> No operation or read transaction selected
<b>RD</b>	12	rw	<b>Read Signal for Status Debug Trigger</b> This bit determines the state of the RD signal of an FPI Bus transaction for which a signal status debug trigger event is generated (if enabled by DBCNTL.ONBOS3 = 1). 0 <sub>B</sub> Trigger on a single read transfer or read cycle of an atomic transfer selected 1 <sub>B</sub> No operation or write transfer selected

Field	Bits	Type	Description
0	7:5, 11:9, 31:13	r	<b>Reserved</b> Read as 0; should be written with 0.

### BCU Debug Trapped Master Register

#### DBGNTT

##### BCU Debug Trapped Master Register

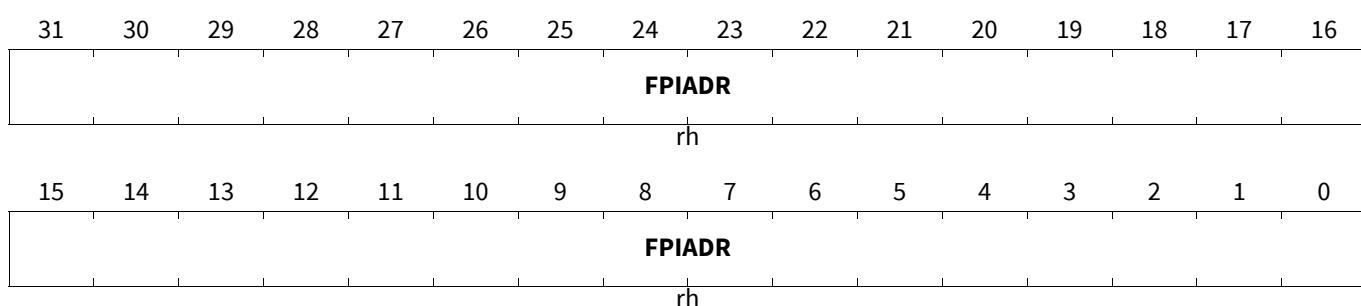
(0044<sub>H</sub>)Debug Reset Value: 0000 FFFF<sub>H</sub>

Field	Bits	Type	Description
FPIGNTi (i=0-15)	i	rh	<b>Master</b> 0 <sub>B</sub> Master i was the FPI bus master. 1 <sub>B</sub> Master i was not the FPI Bus master
0	31:16	rh	<b>Reserved</b> Read as 1 after reset; reading these bits will return the value last written.

### BCU Debug Trapped Address Register

#### DBADRT

##### BCU Debug Trapped Address Register

(0048<sub>H</sub>)Debug Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
FPIADR	31:0	rh	<b>FPI Bus Address Status</b> This register contains the FPI Bus address that was captured when the OCDS break trigger event occurred.

## BCU Debug Trapped Bus Operation Signals Register

### DBBOST

#### BCU Debug Trapped Bus Operation Signals Register(004C<sub>H</sub>)

Debug Reset Value: 0000 3180<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								FPITAG							
r								rh							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>ENDIN IT</b>	<b>FPITO UT</b>	<b>FPIAB ORT</b>	<b>FPIRD</b>	<b>FPIOP S</b>	<b>FPIIRST</b>	<b>FPIWR</b>	<b>FPIRD Y</b>	<b>FPIACK</b>	<b>FPISV M</b>	<b>FPIOPC</b>					
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh			rh	

Field	Bits	Type	Description
<b>FPIOPC</b>	3:0	rh	<p><b>FPI Bus Opcode Status</b>            This bit field indicates the type (opcode) of the FPI Bus transaction captured from the FPI Bus signal lines when the BCU break trigger event occurred.            Other bit combinations are reserved.</p> <ul style="list-style-type: none"> <li>0<sub>H</sub> Single byte transfer</li> <li>1<sub>H</sub> Single half-word transfer</li> <li>2<sub>H</sub> Single word transfer</li> <li>4<sub>H</sub> 2-word block transfer</li> <li>5<sub>H</sub> 4-word block transfer</li> <li>6<sub>H</sub> 8-word block transfer</li> <li>F<sub>H</sub> No operation</li> </ul>
<b>FPI SVM</b>	4	rh	<p><b>FPI Bus Supervisor Mode Status</b>            This bit indicates the state of the Supervisor Mode signal captured from the FPI Bus signal lines when the BCU break trigger event occurred.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> User mode</li> <li>1<sub>B</sub> Supervisor mode</li> </ul>
<b>FPIACK</b>	6:5	rh	<p><b>FPI Bus Acknowledge Status</b>            This bit field indicates the acknowledge signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. Coding see <a href="#">Table 74</a>.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> NSC: No special case</li> <li>01<sub>B</sub> Reserved</li> <li>10<sub>B</sub> RTY: Retry</li> <li>11<sub>B</sub> ERR: Bus Error</li> </ul>
<b>FPI RDY</b>	7	rh	<p><b>FPI Bus Ready Status</b>            This bit indicates the ready signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> Last cycle of transfer</li> <li>1<sub>B</sub> Not last cycle of transfer</li> </ul>

Field	Bits	Type	Description
<b>FPIWR</b>	8	rh	<b>FPI Bus Write Indication Status</b> This bit indicates the write signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 <sub>B</sub> Single write transfer or write cycle of an atomic transfer 1 <sub>B</sub> No operation or read transfer
<b>FPIRST</b>	10:9	rh	<b>FPI Bus Reset Status</b> This bit field indicates the reset signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. Others Reserved 00 <sub>B</sub> Reset of all FPI Bus components 11 <sub>B</sub> No reset
<b>FPIOPS</b>	11	rh	<b>FPI Bus OCDS Suspend Status</b> This bit indicates the OCDS suspend signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 <sub>B</sub> No OCDS suspend request is pending 1 <sub>B</sub> An OCDS suspend request is pending
<b>FPIRD</b>	12	rh	<b>FPI Bus Read Indication Status</b> This bit indicates the read signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 <sub>B</sub> Single read transfer or read cycle of an atomic transfer 1 <sub>B</sub> No operation or write transfer
<b>FPIABORT</b>	13	rh	<b>FPI Bus Abort Status</b> This bit indicates the abort signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 <sub>B</sub> A transfer that has already started was aborted 1 <sub>B</sub> Normal operation
<b>FPIOUT</b>	14	rh	<b>FPI Bus Time-out Status</b> This bit indicates the time-out signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 <sub>B</sub> Normal operation 1 <sub>B</sub> A time-out event was generated
<b>ENDINIT</b>	15	rh	<b>FPI Bus Endinit Status</b> This bit indicates the ENDINIT signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 <sub>B</sub> Normal operation 1 <sub>B</sub> System was in ENDINIT state
<b>FPITAG</b>	21:16	rh	<b>FPI Bus Master TAG Status</b> This bit field indicates the master TAG captured from the FPI Bus signal lines when the BCU break trigger event occurred (see <a href="#">Table 86</a> ). The master TAG identifies the master of the transfer which generated BCU break trigger event.
<b>0</b>	31:22	r	<b>Reserved</b> Read as 0; should be written with 0.

## BCU Debug Data Status Register

### DBDAT

**BCU Debug Data Status Register** **(0050<sub>H</sub>)** **Debug Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>FPIDATA</b>															
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>FPIDATA</b>															
rh															

Field	Bits	Type	Description
<b>FPIDATA</b>	31:0	rh	<b>FPI Bus Data Status</b> This register contains the FPI Bus data that was captured when the OCDS break trigger event occurred.

## BCU EDC Alarm Status Register x

The BCU provides one Alarm Status Register bit for each implemented FPI master and FPI slave.

### ALSTATx (x=0-3)

**BCU EDC Alarm Status Register x** **(0060<sub>H</sub>+x\*4)** **Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>AL31</b>	<b>AL30</b>	<b>AL29</b>	<b>AL28</b>	<b>AL27</b>	<b>AL26</b>	<b>AL25</b>	<b>AL24</b>	<b>AL23</b>	<b>AL22</b>	<b>AL21</b>	<b>AL20</b>	<b>AL19</b>	<b>AL18</b>	<b>AL17</b>	<b>AL16</b>
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>AL15</b>	<b>AL14</b>	<b>AL13</b>	<b>AL12</b>	<b>AL11</b>	<b>AL10</b>	<b>AL09</b>	<b>AL08</b>	<b>AL07</b>	<b>AL06</b>	<b>AL05</b>	<b>AL04</b>	<b>AL03</b>	<b>AL02</b>	<b>AL01</b>	<b>AL00</b>
rh															

Field	Bits	Type	Description
<b>ALy (y=00-31)</b>	y	rh	<b>Alarm y</b> The Alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave / Master interface. 0 <sub>B</sub> No error was detected 1 <sub>B</sub> An error was detected

## BCU EDC Alarm Clear Register x

The BCU provides one Alarm Clear Register bit for each implemented FPI master and FPI slave.

**ALCLRx (x=0-3)****BCU EDC Alarm Clear Register x**(0070<sub>H</sub>+x\*4)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLR31	CLR30	CLR29	CLR28	CLR27	CLR26	CLR25	CLR24	CLR23	CLR22	CLR21	CLR20	CLR19	CLR18	CLR17	CLR16
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLR15	CLR14	CLR13	CLR12	CLR11	CLR10	CLR09	CLR08	CLR07	CLR06	CLR05	CLR04	CLR03	CLR02	CLR01	CLR00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Field	Bits	Type	Description
CL Ry (y=00-31)	y	w	<p><b>Clear alarm y</b></p> <p>The Alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave / Master interface.</p> <p>0<sub>B</sub> No action 1<sub>B</sub> Clears related SBC_ALSTATx.[y] (clear = set to '0'); read always returns 0</p>

**BCU EDC Alarm Control Register****ALCTRL****BCU EDC Alarm Control Register**(0080<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								0							
								r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							0								
							r								
														ALOV C	ALOV
														w	rh

Field	Bits	Type	Description
ALOV	0	rh	<p><b>Alarm Overflow</b></p> <p>The ALOV bit is set if multiple FPI EDC alarms for the same FPI slave or the same FPI master were detected while the related ALSTATx[y] bit was still set.</p> <p><b>Note:</b> Some errors result in a static fault situation, for example address phase, data phase or data enable signal faults. Static faults do not generate multiple alarms and will not set the ALOV bit.</p> <p>0<sub>B</sub> No Alarm Overflow for any FPI EDC alarm detected 1<sub>B</sub> Alarm Overflow detected for at least one of the set ALSTATx[y] bits</p>

Field	Bits	Type	Description
<b>ALOVCLR</b>	1	w	<p><b>Alarm Overflow Clear</b></p> <p>The ALOVCLR bit is required to reset the ALOV bit.</p> <p><math>0_B</math> No action</p> <p><math>1_B</math> Clear ALOV (clear = set to 0); bit value is not stored; read always returns 0</p>
<b>0</b>	31:2	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

## FPI Error Generation Control Register

The FEGEN register controls the injection of errors in the FPI Error Detection Code mechanism.

Errors will be injected in the FPI EDC related encoders and output enable checks in:

- BCU FPI Slave related Slave Encoder Decoder Module (SEDM)
  - CPU0 FPI Master related Master Encoder Decoder Module (MEDM)
  - BCU FPI TIMEOUT signal
  - BCU FPI GRANT signals
  - All groups of FPI Slave / Master output enable signals that are checked in the BCU

FEGEN

**EPI Error Generation Control Register** (0084..) **Application Reset Value:** 0000 0000..

Memory Map Diagram:

Address	Region	Permissions
31 - 28	<b>BCU</b>	r
27 - 24	<b>BCU</b>	rw
23 - 20	<b>BCU</b>	r
19 - 18	<b>EN</b>	rw
17 - 16	<b>EN</b>	
15 - 12	<b>MEDM</b>	r
11 - 8	<b>MEDM</b>	rw
7 - 4	<b>MEDM</b>	r
3 - 2	<b>SEDM</b>	rw
1 - 0	<b>SEDM</b>	

Field	Bits	Type	Description
<b>SEDM</b>	2:0	rw	<p><b>SEDM (Slave Encoder)</b></p> <p>The errors are injected in the FPI EDC encoders related to the BCU (SPB).</p> <p>Other bit combinations are reserved and do not inject errors.</p> <p><math>011_B</math> Slave Data Phase (disable inversion of the FPI_ACK[0] signal)</p> <p><math>101_B</math> Slave Read Data (ECC LSB flip)</p>
<b>MEDM</b>	10:8	rw	<p><b>MEDM (Master Encoder) Type of Error</b></p> <p>The errors are injected in the FPI EDC encoders related to CPU0 (SPB).</p> <p>Other bit combinations are reserved and do not inject errors.</p> <p><math>001_B</math> Master REQUEST Signal (disable REQUEST signal inversion)</p> <p><math>010_B</math> Master LOCK Signal (disable LOCK signal inversion)</p> <p><math>011_B</math> Master Address Phase (invert ECC LSB)</p> <p><math>101_B</math> Master Data Phase (disable inversion of the FPI_ACK[0] signal, Default Master)</p> <p><math>110_B</math> Master Write Data (invert ECC LSB)</p> <p><math>111_B</math> Master ABORT Signal (disable ABORT signal inversion)</p>

Field	Bits	Type	Description
<b>EN</b>	18:16	rw	<b>Enable Signal Type of Error</b> The enable signals errors are injected by inverting the signal. For each group of enable signals that is checked ( $> 1$ signal active = error) the enable signal related to BCU / CPU0 is inverted. Other bit combinations are reserved and do not inject errors. $011_B$ Address Phase Enable (A_EN_N, Master) $101_B$ Data Phase Enable (ABORT_EN_N, Master) $110_B$ Data Enable (D_EN_N, Master and Slave) $111_B$ Data Phase Output Enable (ACK_EN_N, Master and Slave)
<b>BCU</b>	25:24	rw	<b>BCU Type of Error</b> Other bit combinations are reserved and do not inject errors. $10_B$ BCU Grant to CPU0 (disable inversion of the CPU0 Grant signal) $11_B$ BCU Timeout (disable BCU TIMEOUT signal inversion)
<b>0</b>	7:3, 15:11, 23:19, 31:26	r	<b>Reserved</b> Read as 0; should be written with 0.

#### 4.10.6.2 System Registers

Figure 44 shows these system registers which include registers for:

The BCU module includes the following AURIX™ TC3xx Platform standard system registers

- Register access protection

The following standard system registers are not included:

- Module Clock Control
- Module Kernel Reset
- OCDS Control and Status Register

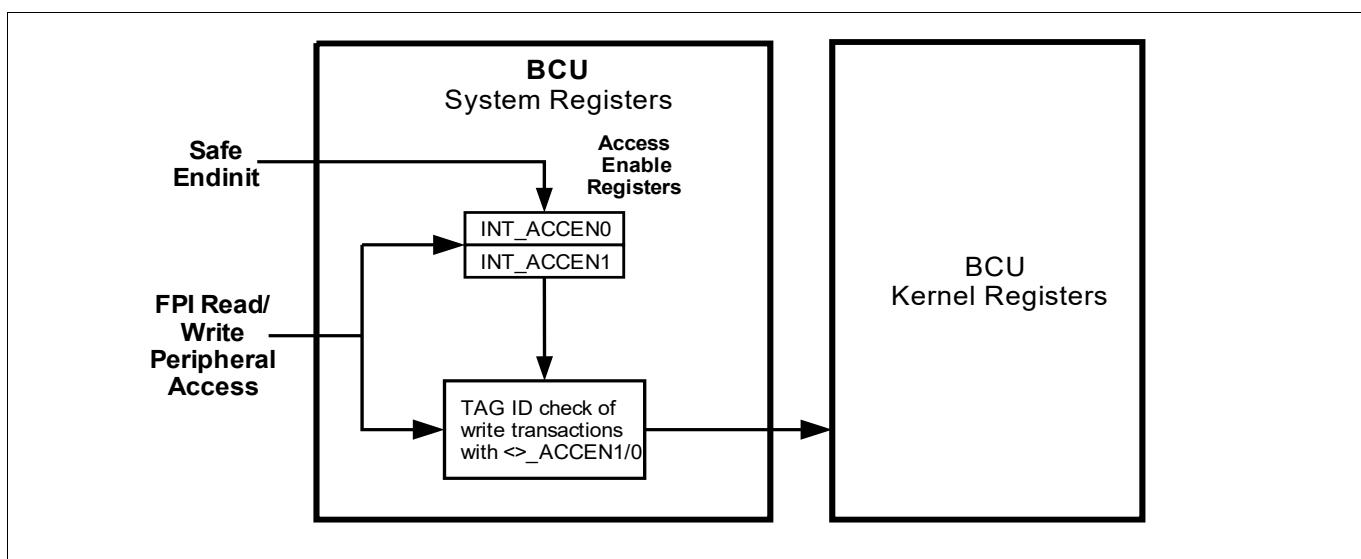


Figure 44 FPI, BCU Implemented System Registers

#### 4.10.6.3 Register Access Protection (ACCEN1/0)

The module provides a master TAG ID based write access protection as part of the safety concept. Each on chip resource with direct or indirect bus master capability has a unique master TAG ID that can be used to identify the master of an on chip bus transaction (see also chapter On Chip Bus Systems).

The SRC register is write protected via an On Chip Bus Master TAG-ID protection (see [Chapter 4.10.6.3](#)). This protection is controlled via the Interrupt Router control registers ACCEN10 and ACCEN00.

TAG ID based protection means that the support of write transactions to the FPI, BCU control registers can be enabled / disabled for each master TAG ID individually. For a disabled master TAG ID, write access will be disconnected with error acknowledge, read access will be processed (see also [Figure 44](#)).

The register access protection is controlled via the registers INT\_ACCEN1 and INT\_ACCEN0 where each bit is related to one encoding of the 6 bit On Chip Master TAG ID.

The INT\_ACCEN1/0 registers are controlling the write access to all module control. The system registers INT\_ACCEN1/0 are Safety Endinit protected.

After reset, all access enable bits and access control bits are enabled, access protection mechanism has to be configured and checked to bring the system in a safe state.

##### Access Enable Register 1

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). ACCEN11/01 are not implemented with register bits as the related On Chip Bus Master TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000B, EN1 -> TAG ID 100001B, ... ,EN31 -> TAG ID 111111B.

##### ACCEN1

###### Access Enable Register 1

(00F8<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							0								
								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							0								
								r							

Field	Bits	Type	Description
0	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

##### Access Enable Register 0

The Access Enable Register 0 controls write access for transactions with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The registers ACCEN00 / ACCEN01 are providing one enable bit for each 6-bit On Chip Bus Master TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ... ,EN31 -> TAG ID 011111B.

## ACCENO

### Access Enable Register 0

(00FC<sub>H</sub>)

Application Reset Value: FFFF FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN31</b>	<b>EN30</b>	<b>EN29</b>	<b>EN28</b>	<b>EN27</b>	<b>EN26</b>	<b>EN25</b>	<b>EN24</b>	<b>EN23</b>	<b>EN22</b>	<b>EN21</b>	<b>EN20</b>	<b>EN19</b>	<b>EN18</b>	<b>EN17</b>	<b>EN16</b>
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EN15</b>	<b>EN14</b>	<b>EN13</b>	<b>EN12</b>	<b>EN11</b>	<b>EN10</b>	<b>EN9</b>	<b>EN8</b>	<b>EN7</b>	<b>EN6</b>	<b>EN5</b>	<b>EN4</b>	<b>EN3</b>	<b>EN2</b>	<b>EN1</b>	<b>ENO</b>
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

#### 4.10.6.4 Kernel Reset Registers (KRST1/0, KRSTCLR)

The BCU does not support the module kernel reset feature and does include the kernel reset registers (KRST1, KRST0, KRSTCLR).

#### 4.10.6.5 Clock Control Register (CLC)

The FPI, BCU module does not include the module clock control (CLC).

*Note:* The FPI, BCU module does not support the Clock Control register functionality which means that the FPI, BCU module clock can not be disabled by the CLC register.

#### 4.10.6.6 OCDS Control and Status Register (OCS)

The FPI, BCU module does not include OCDS Control and Status (OCS) register.

*Note:* The module does not support the OCS register functionality.

#### 4.10.7 On Chip Bus Master TAG Assignments

Each master interface on the System Peripheral Bus and on the SRI Bus is assigned to a 6-bit identification number, the master TAG number (see **Table 86**). This makes it possible for software debug and MCDS purposes to distinguish which master has performed the current transaction.

**Table 86 On Chip Bus Master TAG Assignments**

TAG-Number	Module	Location	Visibility	Description
000000 <sub>B</sub>	DMA	SRI/SPB	SRI/SPB/BBB	DMA Resource Partition 0
000001 <sub>B</sub>	CPU0	SRI/SPB	SRI/SPB/BBB	DMI - Non Safe TAG ID
000010 <sub>B</sub>	CPU0	SRI/SPB	SRI/SPB/BBB	DMI - Safe TAG ID
000011 <sub>B</sub>	HSM	SPB	SRI/SPB/BBB	HSMCMI, HSMRMI <sup>1)</sup>

**Table 86 On Chip Bus Master TAG Assignments (cont'd)**

<b>TAG-Number</b>	<b>Module</b>	<b>Location</b>	<b>Visibility</b>	<b>Description</b>
000100 <sub>B</sub>	DMA	SRI/SPB	SRI/SPB/BBB	DMA Resource Partition 1
000101 <sub>B</sub>	CPU1	SRI/SPB	SRI/SPB/BBB	DMI - Non Safe TAG ID
000110 <sub>B</sub>	CPU1	SRI/SPB	SRI/SPB/BBB	DMI - Safe TAG ID
000111 <sub>B</sub>	SDMMC	SPB	SRI/SPB	SDMMC
001000 <sub>B</sub>	DMA	SRI/SPB	SRI/SPB/BBB	DMA Resource Partition 2
001001 <sub>B</sub>	CPU2	SRI/SPB	SRI/SPB/BBB	DMI / DMBI - Non Safe TAG ID
001010 <sub>B</sub>	CPU2	SRI/SPB	SRI/SPB/BBB	DMI / DMBI - Safe TAG ID
001011 <sub>B</sub>	HSSL0	SRI/SPB	SRI/SPB/BBB	High Speed Serial Link 0
001100 <sub>B</sub>	DMA	SRI/SPB	SRI/SPB/BBB	DMA Resource Partition 3
001101 <sub>B</sub>	CPU3	SRI/SPB	SRI/SPB/BBB	DMI / DMBI - Non Safe TAG ID
001110 <sub>B</sub>	CPU3	SRI/SPB	SRI/SPB/BBB	DMI / DMBI - Safe TAG ID
001111 <sub>B</sub>	HSSL1	SRI/SPB	SRI/SPB/BBB	High Speed Serial Link 1
010000 <sub>B</sub>	-	-	-	Reserved
010001 <sub>B</sub>	CPU4	SRI/SPB	SRI/SPB/BBB	DMI / DMBI.Non Safe TAG ID
010010 <sub>B</sub>	CPU4	SRI/SPB	SRI/SPB/BBB	DMI / DMBI.Safe TAG ID
010011 <sub>B</sub>	CIF1	BBB	BBB	CIF
010100 <sub>B</sub>	-	-	-	Reserved
010101 <sub>B</sub>	CPU5	SRI/SPB	SRI/SPB/BBB	DMI / DMBI - Non Safe TAG ID
010110 <sub>B</sub>	CPU5	SRI/SPB	SRI/SPB/BBB	DMI / DMBI - Safe TAG ID
010111 <sub>B</sub>	-	-	-	Reserved
011000 <sub>B</sub>	-	-	-	Reserved
011001 <sub>B</sub>	-	-	-	Reserved
011010 <sub>B</sub>	-	-	-	Reserved
011011 <sub>B</sub>	-	-	-	Reserved
011100 <sub>B</sub>	Cerberus	SRI/SPB	SRI/SPB/BBB	Cerberus (DAP, JTAG)
011100 <sub>B</sub>	IOC32P	BBB	BBB	IOC32P (DAP, JTAG)
011101 <sub>B</sub>	-	-	-	Reserved
011110 <sub>B</sub>	IOC32E	BBB	BBB	IOC32E (DAPE)
011111 <sub>B</sub>	-	-	-	Reserved
100000 <sub>B</sub>	CPU0	SRI	SRI	PMI / PMBI
100001 <sub>B</sub>	CPU1	SRI	SRI	PMI / PMBI
100010 <sub>B</sub>	CPU2	SRI	SRI	PMI / PMBI
100011 <sub>B</sub>	CPU3	SRI	SRI	PMI / PMBI
100100 <sub>B</sub>	CPU4	SRI	SRI	PMI / PMBI
100101 <sub>B</sub>	CPU5	SRI	SRI	PMI / PMBI
100110 <sub>B</sub>	-	-	-	Reserved
100111 <sub>B</sub>	-	-	-	Reserved

**Table 86 On Chip Bus Master TAG Assignments (cont'd)**

<b>TAG-Number</b>	<b>Module</b>	<b>Location</b>	<b>Visibility</b>	<b>Description</b>
101000 <sub>B</sub>	GETH DMA0	SRI	SRI	GETH master access only on SRI / to system SRAM resources
101001 <sub>B</sub>	GETH DMA1	SRI	SRI	GETH master access only on SRI / to system SRAM resources
101010 <sub>B</sub>	GETH DMA2	SRI	SRI	GETH master access only on SRI / to system SRAM resources
101011 <sub>B</sub>	GETH DMA3	SRI	SRI	GETH master access only on SRI / to system SRAM resources
101100 <sub>B</sub>	GETH1 DMA0	SRI	SRI	GETH1 master access only on SRI / to system SRAM resources
101101 <sub>B</sub>	GETH1 DMA1	SRI	SRI	GETH1 master access only on SRI / to system SRAM resources
101110 <sub>B</sub>	GETH1 DMA2	SRI	SRI	GETH1 master access only on SRI / to system SRAM resources
101111 <sub>B</sub>	GETH1 DMA3	SRI	SRI	GETH1 master access only on SRI / to system SRAM resources
110000 <sub>B</sub>	DAM0	SRI	SRI	DAM0
110001 <sub>B</sub>	DAM1	SRI	SRI	DAM1
111111 <sub>B</sub>	-	-	-	Reserved Not used by any Master Interface for bus access. Note: this encoding is used to disable an individual FPB in the PFI.
Others	-	-	-	Reserved

1) Both HSM FPI Master Interfaces (HSMMCMI, HSMMRMI) are using the same TAG ID

#### 4.10.8 Revision History

**Table 87 Revision History**

<b>Reference</b>	<b>Change to Previous Version</b>	<b>Comment</b>
V1.2.7	No functional change.	
V1.2.8	No functional changes.	-

---

**CPU Subsystem**

## 5 CPU Subsystem

This chapter describes the implementation-specific options of the TC1.6.2P TriCore CPUs found in the AURIX™ series of devices.

This chapter should be read in conjunction with the TriCore Architecture Manual. Topics covered by the architecture manual include:-

- Architectural Overview
- Programming Model
- CPU Registers
- Tasks and Functions
- Interrupt Handling
- Traps
- Memory Protection System
- Temporal Protection System
- Floating Point Operations
- Debug
- Instruction Set

## CPU Subsystem

### 5.1 Feature List

Key CPU Features include:

#### Architecture

- 32-bit load store architecture
- 4 Gbyte address range ( $2^{32}$ )
- 16-bit and 32-bit instructions for reduced code size
- Data types:
  - Boolean, integer with saturation, bit array, signed fraction, character, double-word integers, signed integer, unsigned integer, IEEE-754 single-precision floating point
- Data formats:
  - Bit, byte (8-bits), half-word (16-bits), word (32-bits), double-word (64-bits)
- Byte and bit addressing
- Little-endian byte ordering for data, memory and CPU registers
- Multiply and Accumulate (MAC) instructions: Dual  $16 \times 16$ ,  $16 \times 32$ ,  $32 \times 32$
- Saturation integer arithmetic
- Packed data
- Addressing modes:
  - Absolute, circular, bit reverse, long + short, base + offset with pre- and post-update
- Instruction types:
  - Arithmetic, address arithmetic, comparison, address comparison, logical, MAC, shift, coprocessor, bit logical, branch, bit field, load/store, packed data, system
- General Purpose Register Set (GPRS):
  - Sixteen 32-bit data registers
  - Sixteen 32-bit address registers
  - Three 32-bit status and program counter registers (PSW, PC, PCXI)
- Debug support (OCDS):
  - Level 1, supported in conjunction with the CPS block
  - Level 3, supported in conjunction with the MCDS block (Emulation Device only).
- Flexible memory protection system providing multiple protection sets with multiple protection ranges per set.
- Temporal protection system allowing time bounded real time operation.

#### TC1.6.2P Implementation

- Most instructions executed in 1 cycle
- Branch instructions in 1, 2 or 3 cycles (using dynamic branch prediction)
- Wide memory interface for fast context switch
- Automatic context save-on-entry and restore-on-exit for: subroutine, interrupt, trap
- Six memory protection register sets
- Dual instruction issuing (in parallel into Integer Pipeline and Load/Store Pipeline)
- Third pipeline for loop instruction only (zero overhead loop)
- Single precision Floating Point Unit (IEEE-754 Compatible)

---

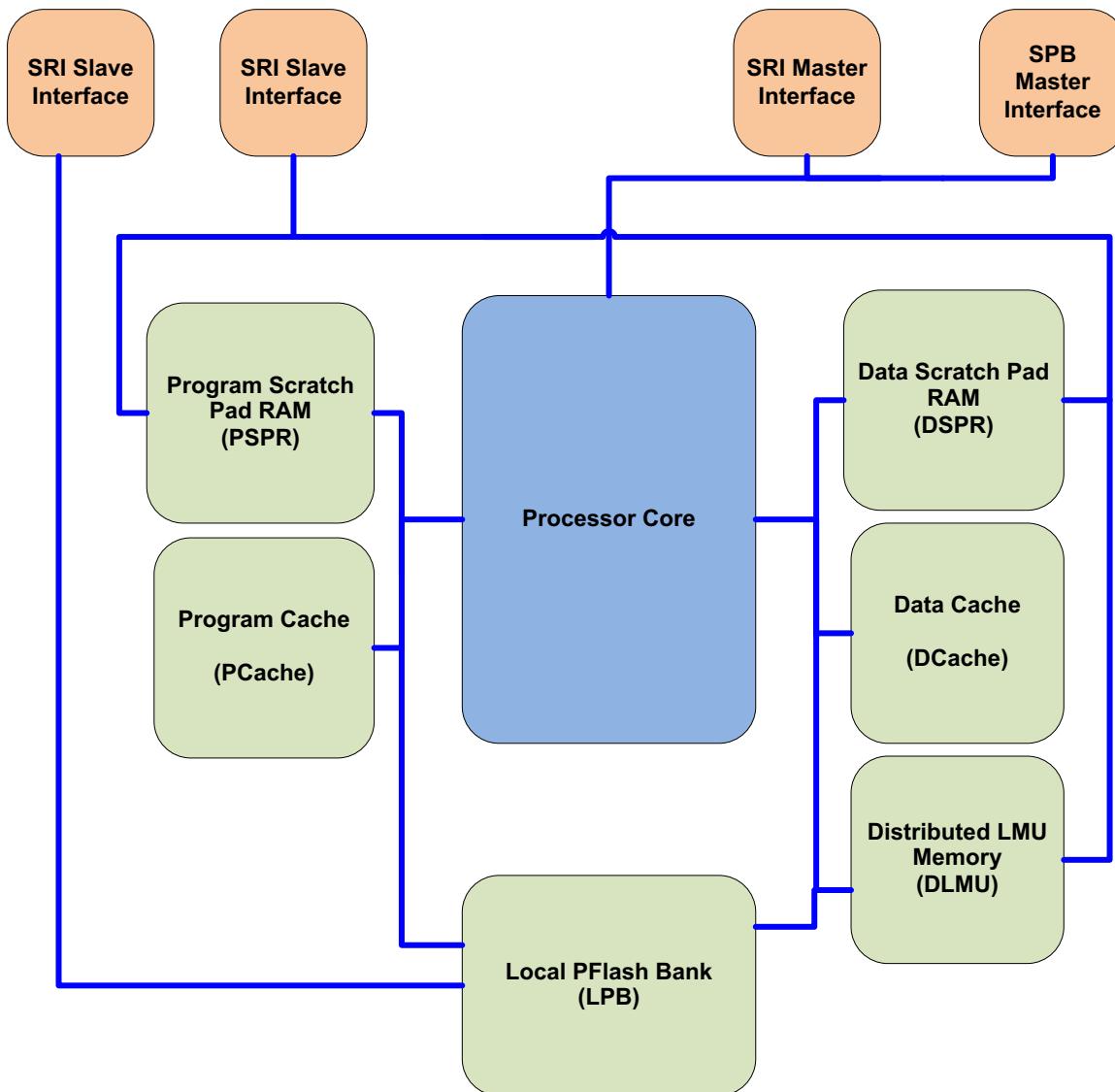
## CPU Subsystem

- Dedicated Integer divide unit
- 18 data memory protection ranges, 10 code memory protection ranges arranged in 6 sets

## CPU Subsystem

### 5.2 Overview

The following sections give an overview of the TC1.6.2P Implementation



**Figure 45 Processor Core, Local Memory and connectivity**

The processor core connects to the following memories and bus interfaces (where implemented)

- Data Scratchpad SRAM (DSPR)
- Program Scratchpad SRAM (PSPR)
- Data Cache (DCache)
- Program Cache (PCache)
- Distributed LMU memory (DLMU)
- Local Pflash bank (LPB)
- SRI slave interface (x2)
- SRI master Interface
- SPB master interface

## CPU Subsystem

### 5.2.1 CPU Diagram

The Central Processing Unit (CPU) comprises of an Instruction Fetch Unit, an Execution Unit, a General Purpose Register File (GPR), a CPU Slave interface (CPS), and Floating Point Unit (FPU).

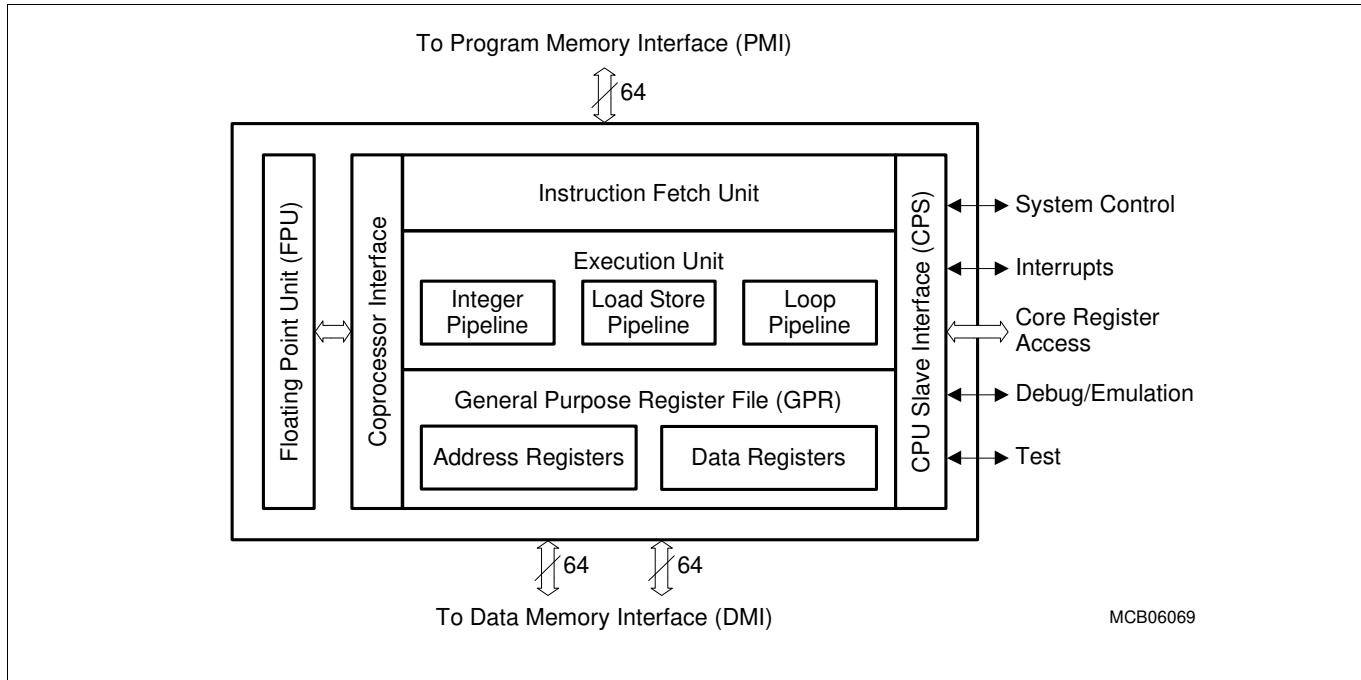


Figure 46 CPU Block Diagram

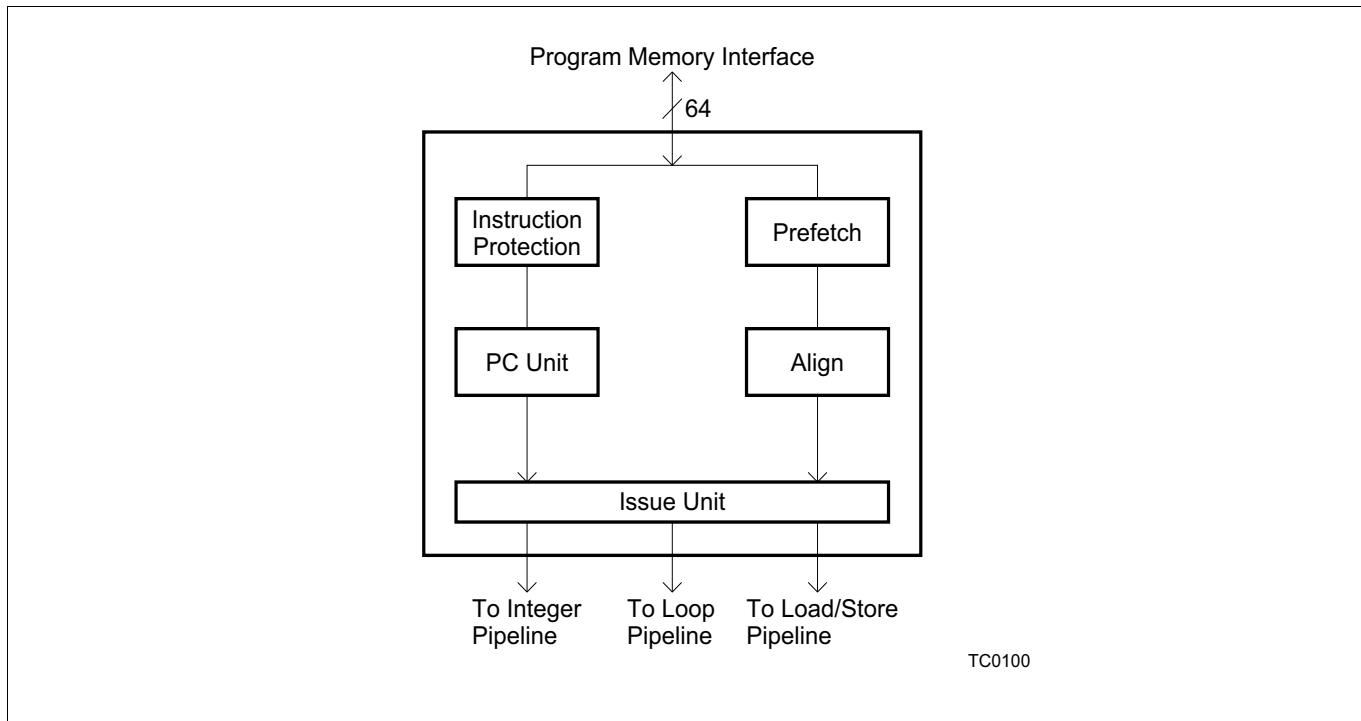
### 5.2.2 Instruction Fetch Unit

The Instruction Fetch Unit pre-fetches and aligns incoming instructions from the 64-bit wide Program Memory Interface (PMI). Instructions are placed in predicted program order in the Issue fifo. The Issue fifo buffers up to six instructions and directs the instruction to the appropriate execution pipeline.

The Instruction Protection Unit checks the validity of accesses to the PMI and the integrity of incoming instructions fetched from the PMI.

The branch unit examines the fetched instructions for branch conditions and predicts the most likely execution path based on previous branch behavior. The Program Counter Unit (PC) is responsible for updating the program counters.

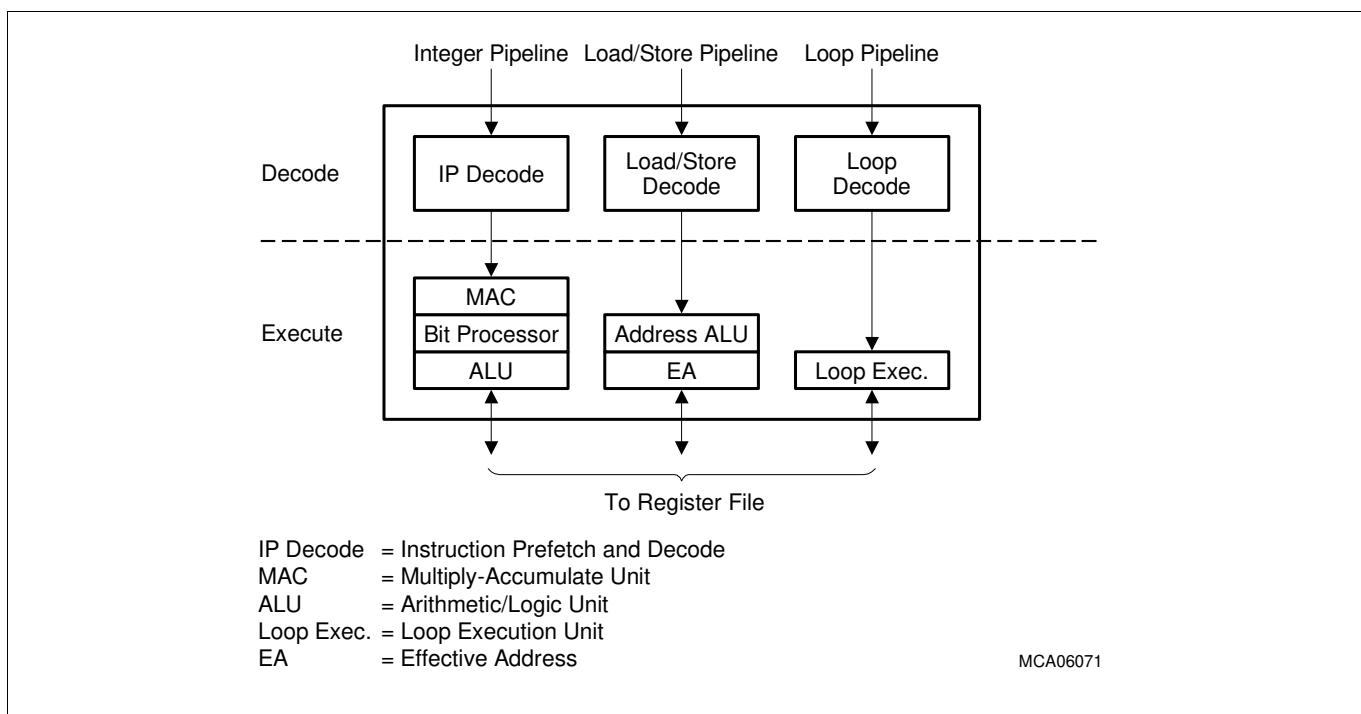
## CPU Subsystem



**Figure 47 Instruction Fetch Unit**

### 5.2.3 Execution Unit

The Execution Unit contains the Integer Pipeline, the Load/Store Pipeline and the Loop Pipeline. All three pipelines operate in parallel, permitting up to three instructions to be executed in one clock cycle. In the execution unit all instructions pass through a decode stage followed by two execute stages. Pipeline hazards (stalls) are minimised by the use of forwarding paths between pipeline stages allowing the results of one instruction to be used by a following instruction as soon as the result becomes available.



**Figure 48 Execution Unit**

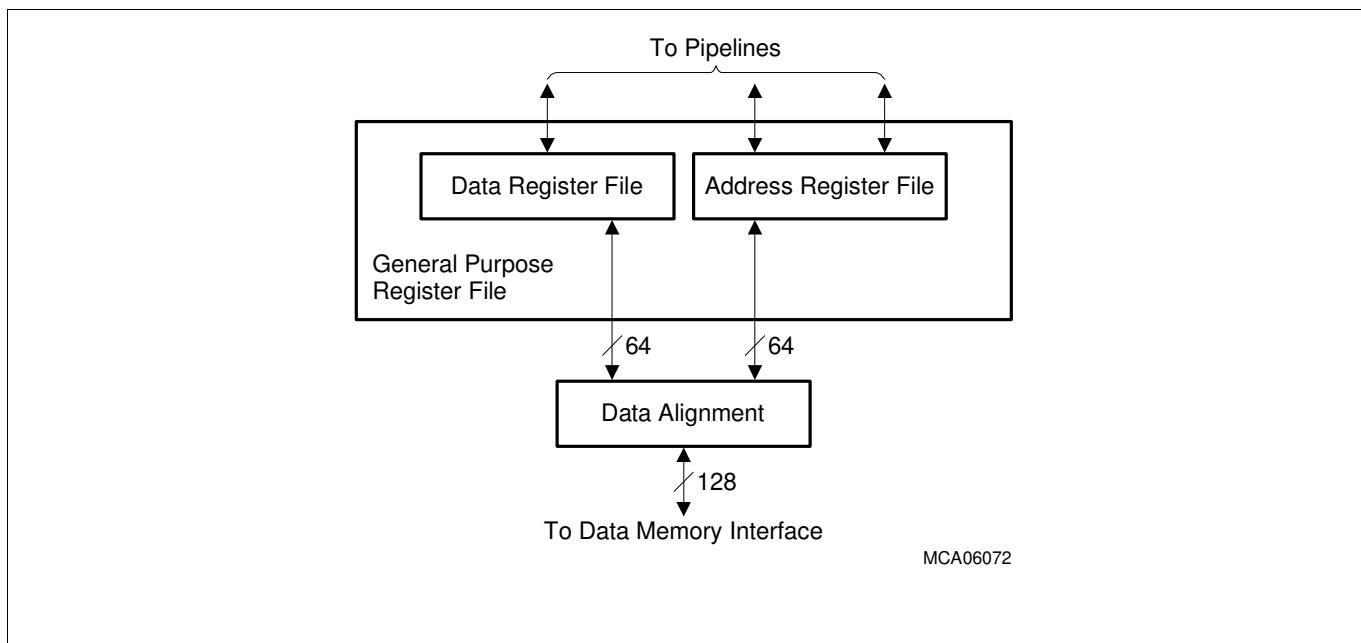
## CPU Subsystem

### 5.2.4 General Purpose Register File

The CPU has a General Purpose Register (GPR) file, divided into an Address Register File (registers A0 through A15) and a Data Register File (registers D0 through D15).

The data flow for instructions issued to the Load/Store Pipeline is steered through the Address Register File.

The data flow for instructions issued to/from the Integer Pipeline and for data load/store instructions issued to the Load/Store Pipeline is steered through the Data Register File.



**Figure 49 General Purpose Register File**

## 5.3 Functional Description

### 5.3.1 Summary of functional changes from AURIX

The TC1.6.2P CPU utilises the same core processing hardware as the TC1.6P processor used in the AURIX series of devices. The main enhancements and changes are listed below:-

- A portion of the LMU memory (called DLMU) is distributed between the processors to provide high performance access to global SRAM
- The PFlash memory is distributed between the processors to provide high performance access to a local PFlash bank. (LPB)
- New instructions (See architecture manual for details):-
  - CRC32B.W, CRC32L.W, CRC32.B (CRC32 for big endian, little endian and byte data)
  - CRCN (arbitrary width and polynomial CRC calculation)
  - SHUFFLE (Reorder bytes within word)
  - POPCNT (count number of bits set in word)
  - FTOHP, HPTOF (Half precision floating point conversion)
  - LHA (Load high bits of address value)
- Enhanced memory protection

## CPU Subsystem

- Number of protection sets increased to 6 (was 4), The PSW.PRS field has been extended to reflect this.
- Number of code protection ranges increased to 10 (was 8)
- Number of data protection ranges increased to 18 (was 16)
- The temporal protection system is extended to provide a dedicated exception timer.
- Independent core resets implemented. (Individual cores may be independently reset as required)
- To exit boot halt the SYSCON.BHALT should be cleared (Was DBGSR.HALT)
- The overlay system is extended to support additional processor cores.
- The store buffer data merge functionality is extended to merge consecutive half words into words and consecutive words into double words.
- The safety protection system has been extended to cover external read and write accesses to local DSPPR/PSPR and DLMU, and to cover external read accesses to the LPB.
- The CPU\_ID has changed (to 0x00C0C020 for TC39X A-Step)

### 5.3.2 Summary of changes from TC39x A-Step

The following changes have been implemented between the TC39x A-step and subsequent devices.

- Extension of ACCEN enable registers to support > 32 bus masters (ACC\*B registers now active)
- Implementation of FLASHCON4.DDIS to provide the ability to disable direct access to Local PFlash.
- Increase in DSPPR sizes for CPU0,1 (96K -> 240K)
- The CPU\_ID has changed to 0x00C0C021
- Emulator space disable bit added to SYSCON register (SYSCON.ESDIS)
- Temporal protection and exception timers are frozen during Suspend.
- Modification to ICU interface to drive invalid ECC values at all times except when there is a valid ACK

**CPU Subsystem****5.3.3 AURIX™ Family CPU configurations**

The different CPU and local memory configurations for the AURIX™ family of devices are detail in the following tables:

**Table 88 Processor and local memory configuration of the TC39x**

Processor	CORE_ID	Program Cache	Program Scratch Pad	Data Cache	Data Scratch Pad	DLMU	LPB	Standby DLMU SRAM	Lock-Step
TC1.6.2P	6	32 KB	64 KB	16 KB	96 KB	64 KB	1 MB	No	No
TC1.6.2P	4	32 KB	64 KB	16 KB	96 KB	64 KB	3 MB	No	No
TC1.6.2P	3	32 KB	64 KB	16 KB	96 KB	64 KB	3 MB	No	Yes
TC1.6.2P	2	32 KB	64 KB	16 KB	96 KB	64 KB	3 MB	No	Yes
TC1.6.2P	1	32 KB	64 KB	16 KB	240 KB	64 KB	3 MB	Yes	Yes
TC1.6.2P	0	32 KB	64 KB	16 KB	240 KB	64 KB	3 MB	Yes	Yes

**Table 89 Processor and local memory configuration of the TC3Ex**

Processor	CORE_ID	Program Cache	Program Scratch Pad	Data Cache	Data Scratch Pad	DLMU	LPB	Standby DLMU SRAM	Lock-Step
TC1.6.2P	3	32 KB	64 KB	16 KB	96 KB	64 KB	3 MB	No	No
TC1.6.2P	2	32 KB	64 KB	16 KB	96 KB	64 KB	3 MB	No	No
TC1.6.2P	1	32 KB	64 KB	16 KB	240 KB	64 KB	3 MB	Yes	Yes
TC1.6.2P	0	32 KB	64 KB	16 KB	240 KB	64 KB	3 MB	Yes	Yes

**Table 90 Processor and local memory configuration of the TC38x**

Processor	CORE_ID	Program Cache	Program Scratch Pad	Data Cache	Data Scratch Pad	DLMU	LPB	Standby DLMU SRAM	Lock-Step
TC1.6.2P	3	32 KB	64 KB	16 KB	96 KB	64 KB	1 MB	No	No
TC1.6.2P	2	32 KB	64 KB	16 KB	96 KB	64 KB	3 MB	No	No
TC1.6.2P	1	32 KB	64 KB	16 KB	240 KB	64 KB	3 MB	Yes	Yes
TC1.6.2P	0	32 KB	64 KB	16 KB	240 KB	64 KB	3 MB	Yes	Yes

**Table 91 Processor and local memory configuration of the TC37xEXT**

Processor	CORE_ID	Program Cache	Program Scratch Pad	Data Cache	Data Scratch Pad	DLMU	LPB	Standby DLMU SRAM	Lock-Step
TC1.6.2P	2	32 KB	64 KB	16 KB	96 KB	64 KB	0 MB	No	Yes
TC1.6.2P	1	32 KB	64 KB	16 KB	240 KB	64 KB	3 MB	Yes	Yes
TC1.6.2P	0	32 KB	64 KB	16 KB	240 KB	64 KB	3 MB	Yes	Yes

**CPU Subsystem****Table 92 Processor and local memory configuration of the TC37x**

<b>Processor</b>	<b>CORE_ID</b>	<b>Program Cache</b>	<b>Program Scratch Pad</b>	<b>Data Cache</b>	<b>Data Scratch Pad</b>	<b>DLMU</b>	<b>LPB</b>	<b>Standby DLMU SRAM</b>	<b>Lock-Step</b>
TC1.6.2P	2	32 KB	64 KB	16 KB	96 KB	64 KB	0 MB	No	No
TC1.6.2P	1	32 KB	64 KB	16 KB	240 KB	64 KB	3 MB	Yes	Yes
TC1.6.2P	0	32 KB	64 KB	16 KB	240 KB	64 KB	3 MB	Yes	Yes

**Table 93 Processor and local memory configuration of the TC36x**

<b>Processor</b>	<b>CORE_ID</b>	<b>Program Cache</b>	<b>Program Scratch Pad</b>	<b>Data Cache</b>	<b>Data Scratch Pad</b>	<b>DLMU</b>	<b>LPB</b>	<b>Standby DLMU SRAM</b>	<b>Lock-Step</b>
TC1.6.2P	1	32 KB	32 KB	16 KB	192 KB	64 KB	2 MB	Yes	Yes
TC1.6.2P	0	32 KB	32 KB	16 KB	192 KB	64 KB	2 MB	Yes	Yes

**Table 94 Processor and local memory configuration of the TC3Ax**

<b>Processor</b>	<b>CORE_ID</b>	<b>Program Cache</b>	<b>Program Scratch Pad</b>	<b>Data Cache</b>	<b>Data Scratch Pad</b>	<b>DLMU</b>	<b>LPB</b>	<b>Standby DLMU SRAM</b>	<b>Lock-Step</b>
TC1.6.2P	3	32 KB	64 KB	16 KB	96 KB	64 KB	0 MB	No	No
TC1.6.2P	2	32 KB	64 KB	16 KB	240 KB	64 KB	0 MB	No	No
TC1.6.2P	1	32 KB	64 KB	16 KB	240 KB	64 KB	2 MB	Yes	Yes
TC1.6.2P	0	32 KB	64 KB	16 KB	240 KB	64 KB	2 MB	Yes	Yes

**Table 95 Processor and local memory configuration of the TC35x**

<b>Processor</b>	<b>CORE_ID</b>	<b>Program Cache</b>	<b>Program Scratch Pad</b>	<b>Data Cache</b>	<b>Data Scratch Pad</b>	<b>DLMU</b>	<b>LPB</b>	<b>Standby DLMU SRAM</b>	<b>Lock-Step</b>
TC1.6.2P	2	32 KB	64 KB	16 KB	96 KB	64 KB	0 MB	No	No
TC1.6.2P	1	32 KB	64 KB	16 KB	240 KB	64 KB	2 MB	Yes	Yes
TC1.6.2P	0	32 KB	64 KB	16 KB	240 KB	64 KB	2 MB	Yes	Yes

**Table 96 Processor and local memory configuration of the TC33xEXT**

<b>Processor</b>	<b>CORE_ID</b>	<b>Program Cache</b>	<b>Program Scratch Pad</b>	<b>Data Cache</b>	<b>Data Scratch Pad</b>	<b>DLMU</b>	<b>LPB</b>	<b>Standby DLMU SRAM</b>	<b>Lock-Step</b>
TC1.6.2P	1	32 KB	64 KB	16 KB	96 KB	64 KB	0MB	No	No
TC1.6.2P	0	32 KB	32 KB	16 KB	192 KB	8 KB	2MB	Yes	Yes

---

**CPU Subsystem****Table 97 Processor and local memory configuration of the TC33x**

Processor	CORE_ID	Program Cache	Program Scratch Pad	Data Cache	Data Scratch Pad	DLMU	LPB	Standby DLMU SRAM	Lock-Step
TC1.6.2P	0	32 KB	8 KB	16 KB	192 KB	8 KB	2MB	Yes	Yes

## CPU Subsystem

### 5.3.4 CPU Implementation-Specific Features

This section describes these implementation-specific features of the TC1.6.2P CPUs. For a complete description of all registers, refer to the TriCore Architecture Manual.

#### 5.3.4.1 Context Save Areas / Context Operations

The CPU uses a uniform context-switching method for function calls, interrupts and traps. In all cases the Upper Context of the task is automatically saved and restored by hardware. Saving and restoring of the Lower Context may be optionally performed by software.

The Context Save Areas (CSA) and addresses targeted by explicit context load/store instructions (e.g. LDLCX) may be placed in DSPR, DLMU or external memory (cached or uncached).

##### CSA Placement in DSPR or DLMU

The actual timing of context operations is dependent upon the placement of the Context Save Areas. Maximum performance is achieved when the Context Save Area is placed in DSPR. In this case all context save and restores operations take four cycles (assuming no external interference). If the CSA is placed in the DLMU context save and restores operations take eight cycles

##### CSA Placement in Cached External Memory

In this case, the timing is also dependent on the state of the Data Cache. The best case Data Cache operation occurs when context saves do not incur a cache line writeback, and context restores hit in the data cache. In this case all context saves and restores take eight cycles.

#### 5.3.4.2 Program Counter (PC) Register

The Program Counter (PC) holds the address of the instruction that is currently fetched and forwarded to the CPU pipelines. The CPU handles updates of the PC automatically.

Software can use the current value of the PC for various tasks, such as performing code address calculations. Reading the PC through software executed by the CPU must only be done with an MFCR instruction. Such a read will return the PC of the MFCR instruction itself. Explicit writes to the PC through an MTCR instruction must not be done due to possible unexpected behavior of the CPU. The PC may be written only when the CPU is halted.

The CPU must not perform Load/Store instructions to the mapped address of the PC in Segment 15. A MEM trap will be generated in such a case. Bit 0 of the PC register is read-only and hard-wired to 0.

#### 5.3.4.3 Store Buffers

To increase performance the TC1.6.2P CPUs implements store buffering to decouple memory write operations from CPU instruction execution. All stores from the CPU are placed in the store buffer prior to being written to local memory or transferred via the bus system. Write data is taken from the store buffers and written to memory when the target memory or bus interface becomes available. In normal operation the CPU will prioritise memory load operations over store operations in order to improve performance unless:-

- The store buffer is full.
- The load is to peripheral space and a store to peripheral space exists in the store buffer. (In order peripheral space access).
- The load or store is part of a read-modify-write operation.

Typically the operation of the store buffer is invisible to the end user. If there is a requirement that data is written to local memory prior to execution of a subsequent instruction then a DSYNC instruction may be used to flush the store buffers. If the data is targeted at either the SRI or the SPB address ranges, then a DSYNC instruction may be used to flush the store buffers followed by a load operation from the targeted slave address range.

---

## CPU Subsystem

To improve performance the store buffer will merge consecutive data values to reduce the number of memory accesses required. The following operations are merged in the store buffer

- Consecutive byte writes to the same half word location.
- Consecutive half-word writes to the same word location.
- Consecutive word writes to the same double word location.

The TC1.6.2P CPU store buffer can hold the data for up to 6 stores operations.

Store buffer operation may be disabled by setting the SMACON.IODT bit. This should not be done in normal execution as it will severely limit performance.

## CPU Subsystem

### 5.3.4.4 Interrupt System

An interrupt request can be generated by the on-chip peripheral units, or it can be generated by external events. Requests can be targeted to any CPU.

The interrupt system evaluates service requests for priority and to identify whether the CPU should receive the request. The highest-priority service request is then presented to the CPU by way of an interrupt.

On taking an interrupt the CPU will vector to a unique PC generated from the interrupt priority number and the Base Interrupt Vector (BIV). The spacing between the vector PCs in the interrupt vector table may be selected to be either 32 Bytes or 8 Bytes using BIV[0].

The TC1.6.2P implements a fast interrupt system. This system avoids unnecessary context save and restore operations and hence speeds up interrupt routine entry. A fast interrupt is triggered when:-

- An Interrupt is pending
- A Return From exception instruction is being executed (RFE)
- The priority of the pending interrupt is greater than the priority level that would be returned to should the RFE be executed (ICR.PIPN > PCXI.PCPN).
- Interrupts will be enabled should the RFE be executed. (PCXI.PIE == 1)
- Fast Interrupts are not otherwise disabled due to the presence of MTCR instructions or context operations in the pipeline.

*Note: Only MTCR operations to the following registers will disable fast interrupt PCXI, PCX, PSW, PC, SYSCON, CPU\_ID, CORE\_ID, BIV, BTV, ISP, ICR, FCX, LCX, DMS, DCX, DBGTCR.*

Without the fast interrupt operation the RFE would cause a restore of the upper context immediately followed by a save of the same upper context back to exactly the same memory location (Minimum 8 cycles). The fast interrupt system replaces this redundant restore/save sequence with a load of PCXI/PSW/A10/A11 (Minimum 1 Cycle) from the saved context. System state at the end of the fast interrupt is the same as if the standard RFE/Interrupt sequence had been performed.

### 5.3.4.5 Trap System

The following traps have implementation-specific properties.

#### UOPC - Unimplemented Opcode (TIN 2)

The UOPC trap is raised on optional MMU instructions, coprocessor two and coprocessor three instructions.

#### OPD - Invalid Operand (TIN 3)

The CPU raised OPD traps for instructions that take even-odd register pairs as an operand where if the operand specifier is odd.

#### DSE - Data Access Synchronous Error (TIN 2)

The Data Access Synchronous Bus Error (DSE) trap is generated by the DMI module when a load access from the CPU encounters certain error conditions, such as a Bus error, or an out-of-range access to DSPr. When a DSE trap is generated, the exact cause of the error can be determined by reading the Data Synchronous Trap Register, DSTR. For details of possible error conditions and the corresponding flag bits in DSTR, see [Table “CPUx Data Synchronous Trap Register” on Page 91](#).

## CPU Subsystem

### **DAE - Data Access Asynchronous Error (TIN 3)**

The Data Access Asynchronous Error Trap (DAE) is generated by the DMI module when a store or cache management access from the CPU encounters certain error conditions, such as an Bus error. When a DAE trap is generated, the exact cause of the error can be determined by reading the Data Asynchronous Trap Register, DATR. For details of possible error conditions and the corresponding flag bits in DATR, see [Table “CPUx Data Asynchronous Trap Register” on Page 92](#).

### **PIE Program Memory Integrity Error (TIN 5)**

The PIE trap is raised whenever an uncorrectable memory integrity error is detected in an instruction fetch from a local memory or the SRI bus. The trap is synchronous to the erroneous instruction. The trap is of Class-4 and has a TIN of 5.

Program memories are protected from memory integrity errors on a 64 bit basis. A non-inhibited PIE trap is raised when an attempt is made to execute an instruction from any fetch group containing a memory integrity error.

The PIEAR and PIETR registers may be interrogated to determine the source of any error more precisely. PIE traps are inhibited if PIETR.IED is set.

### **DIE Data Memory Integrity Error (TIN 6)**

The DIE trap is raised whenever an uncorrectable memory integrity error is detected in a data access to a local memory or the SRI bus. The trap is of Class-4 and has a TIN of 6.

DIE traps are always asynchronous independent of the operation which encountered the error.

A DIE trap is raised if any memory half word (local memory) or double word (SRI bus) accessed by a load/store operation contains an uncorrectable error and no other DIE trap have been raised since the clearing of the DIETR.IED bit. The DIEAR and DIETR registers may be interrogated to determine the source of any error more precisely. Subsequent DIE traps are inhibited if DIETR.IED is set.

### **MPX Memory Protection Execute (TIN 4)**

The MPX trap is raised whenever a program attempts to execute an instruction from a memory area for which it does not have execute permission and memory protection is enabled. The trap is of Class-1 and has a TIN of 4.

The TC1.6.2P compares the 64bit aligned fetch group address with the range(s) defined by the memory protection system to determine whether or not execution is permitted.

### **5.3.4.6 WAIT Instruction**

The WAIT instruction will suspend execution until the occurrence of one of the following events.

- Enabled Interrupt
- Non-Maskable Interrupt
- Asynchronous Trap
- Idle Request
- Suspend Request
- Asynchronous Debug Halt or Trap Request

### **5.3.4.7 Invalid Opcode**

The opcode 0x36 is invalid for all TC1.6.2P CPUs in Infineon AURIX™ devices. Unused program memory filled with 0x36363636 etc will generate a IOPC (Invalid Opcode) trap if execution is attempted.

## CPU Subsystem

### 5.3.4.8 Speculation extent

The TC1.6.2 CPU may perform both **necessary** and **speculative** accesses.

- **Necessary accesses** are those that the CPU is required to make to correctly compute the program
- **Speculative accesses** are those that the CPU may make in order to improve performance either in correct or incorrect anticipation of a necessary access.

The following speculative accesses are performed by the TC1.6.2P CPU

Speculative fetch accesses:- The processor may speculative fetch up to 64 bytes of instruction on the predicted execution path.

Speculative data read accesses:- For cached data locations the processor will read an entire cache containing data read for a required access.

The processor does not perform speculative instruction execution.

The processor does not perform speculative write accesses.

The processor does not perform speculative accesses to peripheral space.

### 5.3.4.9 Instruction Memory Range Limitations

To ensure the processor cores are provided with a constant stream of instructions the Instruction Fetch Units will speculatively fetch instructions from up to 64 bytes ahead of the current PC.

If the current PC is within 64 bytes of the top of an instruction memory the Instruction Fetch Unit may attempt to speculatively fetch instruction from beyond the physical memory range. This may then lead to error conditions and alarms being triggered by the bus and memory systems.

The upper 64 bytes of any memory should therefore not be used for instruction storage. This memory area should be initialised and written as “all zeros”.

### 5.3.4.10 Atomicity of Data Accesses

The data alignment rules along with the number of bus transactions for each access type are detailed in the following tables:

#### Alignment Rules

**Table 98 Alignment rules for non-peripheral space**

Access type	Access size	Alignment of address in memory	Min/Max number of SRI bus transactions
Load, Store Data Register	Byte	Byte ( $1_H$ )	1/1
	Half-Word	2 bytes ( $2_H$ )	1/1
	Word	2 bytes ( $2_H$ )	1/2 *
	Double-Word	2 bytes ( $2_H$ )	1/3 *
Load, Store Address Register	Word	4 bytes ( $4_H$ )	1/1
	Double-Word	4 bytes ( $4_H$ )	1/2 *
SWAP.W, LDMST, CMPSWAP.W, SWAPMSK.W, ST.T	Word	4 bytes ( $4_H$ )	1/1
Context Operations	16 x 32-bit registers	64 bytes ( $40_H$ )	2/2 *

## CPU Subsystem

**Table 99 Alignment rules for peripheral space**

Access type	Access size	Alignment of address in memory	Min/Max Number of SRI bus transactions	Min/Max Number of SPB bus transactions
Load, Store Data Register	Byte	Byte (1 <sub>H</sub> )	1/1	1/1
	Half-Word	2 bytes (2 <sub>H</sub> )	1/1	1/1
	Word	4 bytes (4 <sub>H</sub> )	1/1	1/1
	Double-Word	8 bytes (8 <sub>H</sub> )	1/1	1/1
Load, Store Address Register	Word	4 bytes (4 <sub>H</sub> )	1/1	1/1
	Double-Word	8 bytes (8 <sub>H</sub> )	1/1	1/1
SWAP.W, LDMST, ST.T CMPSWAP.W, SWAPMSK.W	Word	4 bytes (4 <sub>H</sub> )	1/1	1/1
Context Operations	16 x 32-bit registers	Not Permitted	-	-

In the case where a single access results in a single bus transaction atomicity of the data is preserved when viewed from any bus master.

In the case where a single access leads to multiple bus transactions (marked as “\*” in the above tables) then atomicity needs to be considered. In these accesses it is possible for another bus master to read or write the target memory location between the bus transactions required to complete the access.

In the case of word and double word access to the SRI bus the number of bus transactions will be 1 for naturally aligned data values and hence atomicity will be preserved.

The atomicity of data accesses to CPU CSFR registers is not guaranteed if the local processor is performing a MTCR instruction to the same CSFR register.

The instructions SWAP.W, LDMST, CMPSWAP.W, SWAPMSK.W, ST.T are always atomic in operation. They perform a word read followed by a word write to a word aligned memory location. Both read and write access permissions must exist for the word aligned address.

### 5.3.4.11 A11 usage

In normal usage A11 will always contain the target of the next RET or RFE instruction. The processor uses this fact to speculatively load the return target ahead of the execution of the RET/RFE instruction. Code that modifies the A11 (e.g. test code) should be aware that any value stored in A11 may be used as the target of such speculation. If the value in A11 is not a valid address the speculation may lead to error conditions and alarms being triggered by the bus and memory systems.

It is therefore recommended that A11 should only ever contain a valid address value.

### 5.3.4.12 Independent Core Kernel Reset

The CPU core may be reset while the rest of the system remains operational. A module reset can be triggered by the safety management unit (SMU) or by writing ‘1’ into both of the module reset registers CPU\_KRST1.RST and register CPU\_KRST0.RST.

The cause of the last reset to occur is indicated by the CPU\_KRST0.RSTSTAT0/1 bits. The status bits may be cleared by writing CPU\_KRSTCLR.CLR to one.

The CPU\_KRSTCLR, CPU\_KRST1 and CPU\_KRST0 registers are protected by system\_endinit.

Registers related to the Debug Reset (Class 1) and the FLASHCON registers are not influenced by the kernel reset.

---

## CPU Subsystem

### 5.3.4.12.1 Kernel Reset Registers

The following registers control the operation of the kernel reset functionality.

## CPU Subsystem

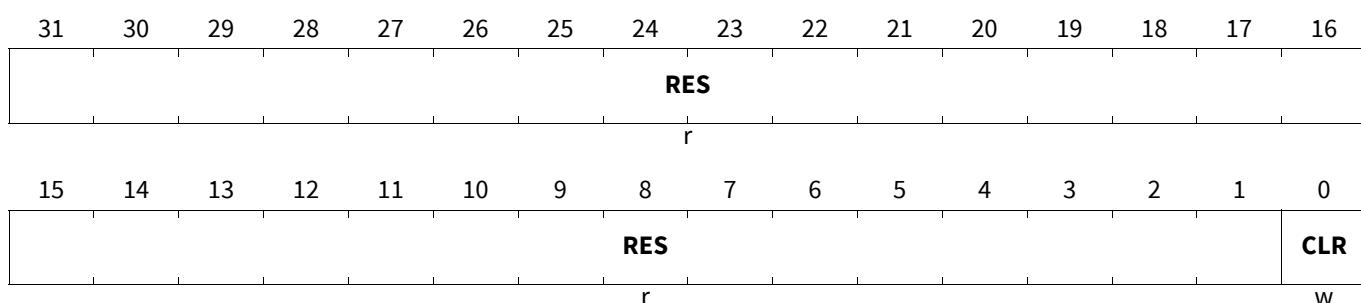
### Register Descriptions

#### CPUx Reset Clear Register

The Kernel Reset Register Clear register is used to clear the Kernel Reset Status bits (CPU\_KRST0.RSTSTAT). This register is protected by system registers ENDINIT.

#### KRSTCLR

**CPUx Reset Clear Register** **(0D008<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
CLR	0	w	<b>Kernel Reset Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear Kernel Reset Status KRST0.RSTSTAT[1:0]
RES	31:1	r	<b>Reserved</b> Read as 0; should be written with 0.

#### CPUx Reset Register 0

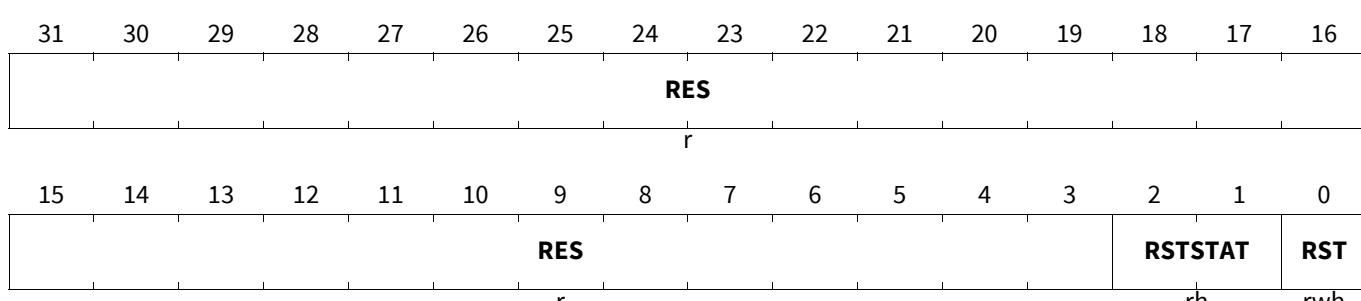
The CPU Kernel Reset Register 0 is used to reset the CPU kernel. CPU registers related to the Debug Reset (Class 1) and the FLASHCON registers are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both CPU Kernel Reset Registers. The RST bits will be re-set at the end of the kernel reset sequence.

CPU Kernel Reset Register 0 includes a kernel reset status bits that is set to '1' at the start of the reset sequence. These bit can be used to detect that a kernel reset was processed. These bits can be re-set to '0' by writing '1' to the related KRSTCLRx.CLR register bit.

This register is protected by system registers ENDINIT.

#### KRST0

**CPUx Reset Register 0** **(0D000<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**



## CPU Subsystem

Field	Bits	Type	Description
<b>RST</b>	0	rwh	<p><b>Kernel Reset</b></p> <p>This reset bit can be used to request a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set.</p> <p>The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.</p> <p>0<sub>B</sub> No kernel reset was requested 1<sub>B</sub> A kernel reset was requested</p>
<b>RSTSTAT</b>	2:1	rh	<p><b>Kernel Reset Status</b></p> <p>This bit indicates whether a kernel reset was executed or not. This bit is set immediately after the execution of a kernel reset.</p> <p>These bits can be cleared by writing '1' to the CLR bit in the related KRSTCLR register.</p> <p>00<sub>B</sub> No kernel reset was executed 01<sub>B</sub> Kernel reset was requested by hardware since last clear (SMU) 10<sub>B</sub> Kernel reset was requested by software since last clear (KRST)</p>
<b>RES</b>	31:3	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

### CPUx Reset Register 1

The CPU Kernel Reset Register 1 is used to reset the CPU kernel. To reset the CPU kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers CPU\_KRSTx1.RST and CPU\_KRSTx0.RST). The RST bit will be re-set at end of the reset sequence. This register is protected by system registers ENDINIT.

#### KRST1

<b>CPUx Reset Register 1</b>																(0D004 <sub>H</sub> )	<b>Application Reset Value: 0000 0000<sub>H</sub></b>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	RES																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	RES																rwh

Field	Bits	Type	Description
<b>RST</b>	0	rwh	<p><b>Kernel Reset</b></p> <p>This reset bit can be used to request a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers are set.</p> <p>The RST bit will be cleared (re-set to '0') after the kernel reset was executed.</p> <p>0<sub>B</sub> No kernel reset was requested 1<sub>B</sub> A kernel reset was requested</p>
<b>RES</b>	31:1	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

---

**CPU Subsystem****5.3.4.13 CPU Clock Control**

The effective CPU execution frequency may be reduced by programming the associated CCUCONn.CPUxDIV register (Where x is the core number). The effective execution frequency seen by the processor is given by the following equation.

(5.1)

$$f_{\text{cpu}} = f_{\text{sri}} \frac{(64 - \text{CPUxDIV})}{64}$$

Where Fcpu is the effective frequency seen by the processor and Fsri is the base SRI frequency. A CPUxDIV value of 0 results in the core being clocked at the SRI frequency (no frequency reduction).

To avoid synchronisation issues typically associated with clock division the clock control mechanism stalls the issue of instructions into the processor pipeline rather than by modifying the actual applied clock. An incoming instruction fetch packet is stalled for the number of cycles required to approximate the required execution frequency. The stall is seen by the processor as a stall in the instruction stream in the same way a stalling instruction memory would be seen.

In most scenarios this mechanism provides a good approximation to clock division based control. The actual reduction in effective frequency will be dependent on the code executed.

## CPU Subsystem

### 5.3.4.14 CPU Core Special Function Registers (CSFR)

This section describes implementation specific features of the Core Special Function Registers.

#### 5.3.4.14.1 Registers

The implementation-specific Program Status Word Register (PSW) is an extension of the PSW description in the TriCore Architecture Manual. The status flags used for FPU operations overlay the status flags used for Arithmetic Logic Unit (ALU) operations.

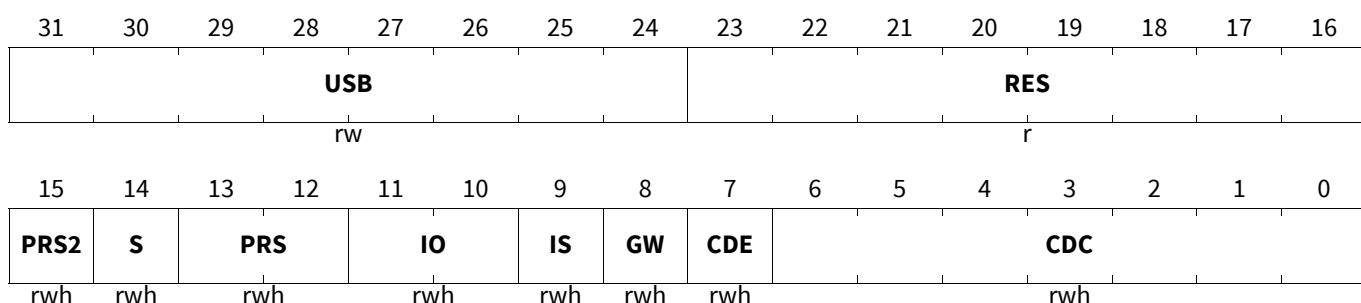
##### CPUx Program Status Word

###### PSW

**CPUx Program Status Word** **(1FE04<sub>H</sub>)** **Application Reset Value: 0000 0B80<sub>H</sub>**

###### CPU\_PSW

**Short address for domain CSFR** **(0FE04<sub>H</sub>)** **Application Reset Value: 0000 0B80<sub>H</sub>**



Field	Bits	Type	Description
CDC	6:0	rwh	<p><b>Call Depth Counter</b></p> <p>Consists of two variable width subfields. The first subfield consists of a string of zero or more initial 1 bits, terminated by the first 0 bit. The remaining bits form the second subfield (CDC.COUNT) which constitutes the Call Depth Count value. The count value is incremented on each Call and is decremented on a Return.</p> <p>0cccccc<sub>B</sub> : 6-bit counter; trap on overflow.</p> <p>10cccccc<sub>B</sub> : 5-bit counter; trap on overflow.</p> <p>110cccccc<sub>B</sub> : 4-bit counter; trap on overflow.</p> <p>1110ccc<sub>B</sub> : 3-bit counter; trap on overflow.</p> <p>11110cc<sub>B</sub> : 2-bit counter; trap on overflow.</p> <p>111110c<sub>B</sub> : 1-bit counter; trap on overflow.</p> <p>1111110<sub>B</sub> : Trap every call (Call Trace mode).</p> <p>1111111<sub>B</sub> : Disable Call Depth Counting.</p> <p>When the call depth count (CDC.COUNT) overflows a trap (CDO) is generated.</p> <p>Setting the CDC to 1111110<sub>B</sub> allows no bits for the counter and causes every call to be trapped. This is used for Call Depth Tracing.</p> <p>Setting the CDC to 1111111<sub>B</sub> disables Call Depth Counting.</p>

## CPU Subsystem

Field	Bits	Type	Description
CDE	7	rwh	<p><b>Call Depth Count Enable</b>            Enables call-depth counting, provided that the PSW.CDC mask field is not all set to 1.            If PSW.CDC = 1111111<sub>B</sub>, call depth counting is disabled regardless of the setting on the PSW.CDE bit.</p> <p>0<sub>B</sub> Call depth counting is temporarily disabled. It is automatically re-enabled after execution of the next Call instruction.</p> <p>1<sub>B</sub> Call depth counting is enabled.</p>
GW	8	rwh	<p><b>Global Address Register Write Permission</b>            Determines whether the current execution thread has permission to modify the global address registers. Most tasks and ISRs use the global address registers as ‘read only’ registers, pointing to the global literal pool and key data structures. However a task or ISR can be designated as the ‘owner’ of a particular global address register, and is allowed to modify it. The system designer must determine which global address variables are used with sufficient frequency and/or in sufficiently time-critical code to justify allocation to a global address register. By compiler convention, global address register A[0] is reserved as the base register for short form loads and stores. Register A[1] is also reserved for compiler use. Registers A[8] and A[9] are not used by the compiler, and are available for holding critical system address variables.</p> <p>0<sub>B</sub> Write permission to global registers A[0], A[1], A[8], A[9] is disabled.</p> <p>1<sub>B</sub> Write permission to global registers A[0], A[1], A[8], A[9] is enabled.</p>
IS	9	rwh	<p><b>Interrupt Stack Control</b>            Determines if the current execution thread is using the shared global (interrupt) stack or a user stack.</p> <p>0<sub>B</sub> User Stack. If an interrupt is taken when the IS bit is 0, then the stack pointer register is loaded from the ISP register before execution starts at the first instruction of the Interrupt Service Routine (ISR).</p> <p>1<sub>B</sub> Shared Global Stack. If an interrupt is taken when the PSW.IS bit is 1, then the current value of the stack pointer is used by the Interrupt Service Routine (ISR).</p>

## CPU Subsystem

Field	Bits	Type	Description
<b>IO</b>	11:10	rwh	<b>Access Privilege Level Control (I/O Privilege)</b> Determines the access level to special function registers and peripheral devices. 00 <sub>B</sub> User-0 Mode No peripheral access. Access to memory regions with the peripheral space attribute are prohibited and results in a PSE or MPP trap. This access level is given to tasks that need not directly access peripheral devices. Tasks at this level do not have permission to enable or disable interrupts. 01 <sub>B</sub> User-1 Mode Regular peripheral access. Enables access to common peripheral devices that are not specially protected, including read/write access to serial I/O ports, read access to timers, and access to most I/O status registers. Tasks at this level may disable interrupts. 10 <sub>B</sub> Supervisor Mode Enables access to all peripheral devices. It enables read/write access to core registers and protected peripheral devices. Tasks at this level may disable interrupts. 11 <sub>B</sub> Reserved
<b>PRS</b>	13:12	rwh	<b>Protection Register Set</b> Selects the active Data and Code Memory Protection Register Set. The memory protection register values control load, store and instruction fetches within the current process. PRS values 111 and 110 are reserved
<b>S</b>	14	rwh	<b>Safe Task Identifier</b>
<b>PRS2</b>	15	rwh	<b>Protection Register Set MSB</b> Selects the active Data and Code Memory Protection Register Set. The memory protection register values control load, store and instruction fetches within the current process. PRS values 111 and 110 are reserved
<b>RES</b>	23:16	r	<b>Reserved</b>
<b>USB</b>	31:24	rw	<b>User Status Bits</b> The eight most significant bits of the PSW are designated as User Status Bits. These bits may be set or cleared as side effects of instruction execution. Refer to the TriCore Architecture manual for details.

## CPUx Previous Context Information Register

### PCXI

**CPUx Previous Context Information Register (1FE00<sub>H</sub>)**

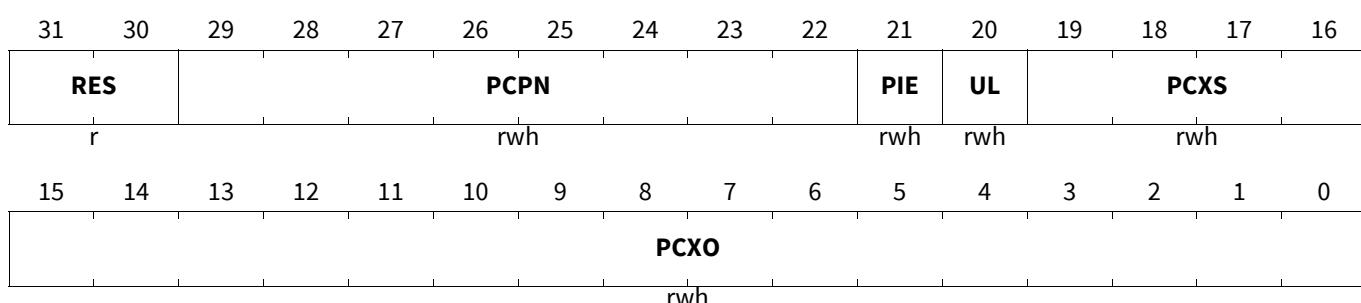
**Application Reset Value: 0000 0000<sub>H</sub>**

### CPU\_PCXI

**Short address for domain CSFR**

(0FE00<sub>H</sub>)

**Application Reset Value: 0000 0000<sub>H</sub>**



## CPU Subsystem

Field	Bits	Type	Description
<b>PCXO</b>	15:0	rwh	<b>Previous Context Pointer Offset Field</b> The PCXO and PCXS fields form the pointer PCX, which points to the CSA of the previous context.
<b>PCXS</b>	19:16	rwh	<b>Previous Context Pointer Segment Address</b> Contains the segment address portion of the PCX. This field is used in conjunction with the PCXO field.
<b>UL</b>	20	rwh	<b>Upper or Lower Context Tag</b> Identifies the type of context saved. If the type does not match the type expected when a context restore operation is performed, a trap is generated. $0_B$ Lower Context $1_B$ Upper Context
<b>PIE</b>	21	rwh	<b>Previous Interrupt Enable</b> Indicates the state of the interrupt enable bit (ICR.IE) for the interrupted task.
<b>PCPN</b>	29:22	rwh	<b>Previous CPU Priority Number</b> Contains the priority level number of the interrupted task.
<b>RES</b>	31:30	r	<b>Reserved</b> Read as 0; should be written as 0.

## CPUx Task Address Space Identifier Register

### TASK\_AS1

CPUx Task Address Space Identifier Register  $(18004_H)$

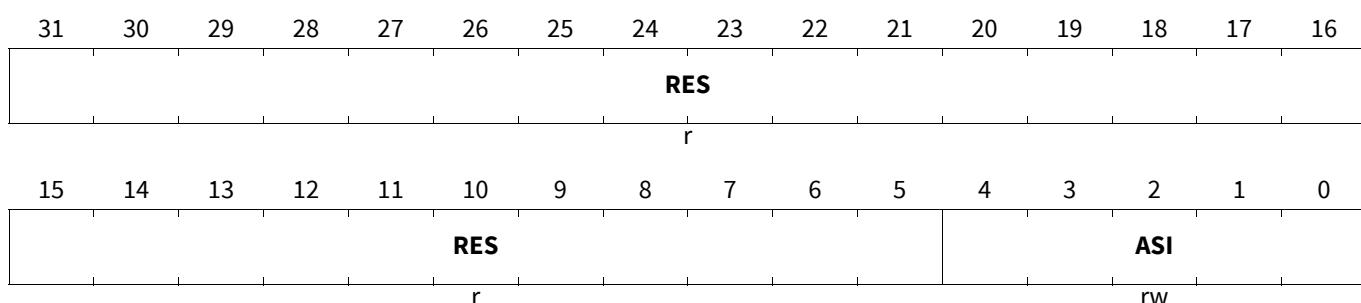
Application Reset Value: 0000 001F<sub>H</sub>

CPU\_TASK\_AS1

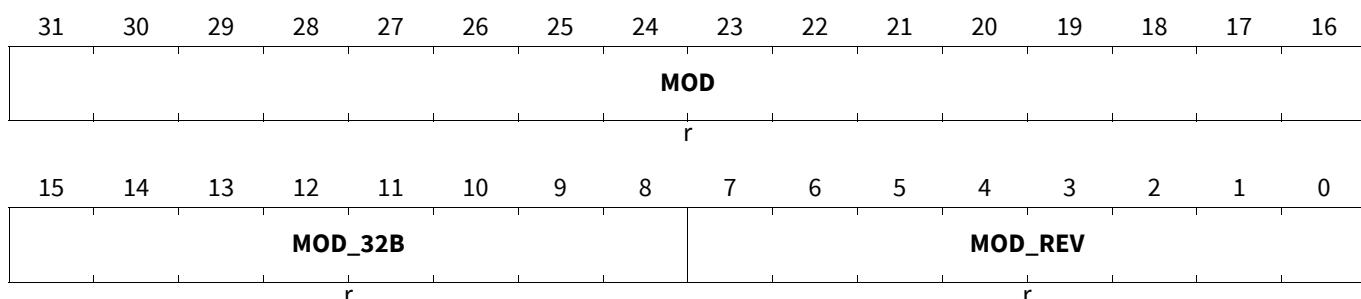
Short address for domain CSFR

$(08004_H)$

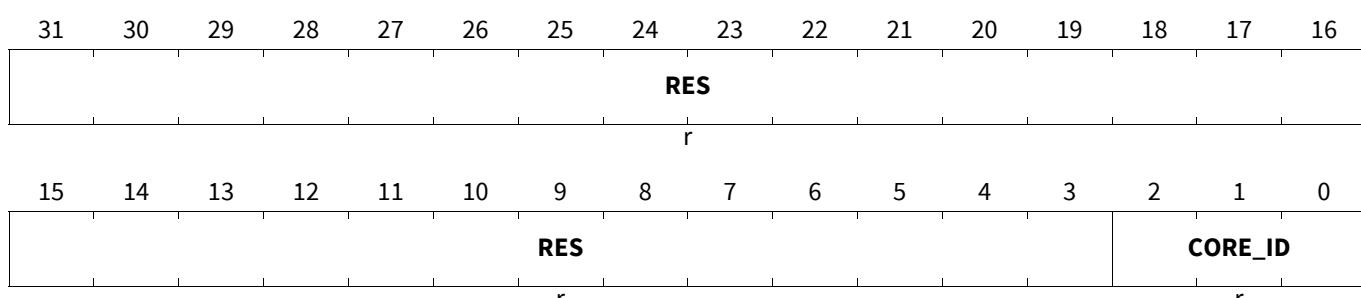
Application Reset Value: 0000 001F<sub>H</sub>



Field	Bits	Type	Description
<b>ASI</b>	4:0	rw	<b>Address Space Identifier</b> The ASI register contains the Address Space Identifier of the current process.
<b>RES</b>	31:5	r	<b>Reserved</b>

**CPU Subsystem****CPUx Identification Register TC1.6.2P****CPU\_ID****CPUx Identification Register TC1.6.2P****(1FE18<sub>H</sub>)****Application Reset Value: 00C0 C021<sub>H</sub>****CPU\_CPU\_ID****Short address for domain CSFR****(0FE18<sub>H</sub>)****Application Reset Value: 00C0 C021<sub>H</sub>**

Field	Bits	Type	Description
<b>MOD_REV</b>	7:0	r	<b>Revision Number</b> 20 <sub>H</sub> Reset value
<b>MOD_32B</b>	15:8	r	<b>32-Bit Module Enable</b> C0 <sub>H</sub> A value of C0 <sub>H</sub> in this field indicates a 32-bit module with a 32-bit module ID register.
<b>MOD</b>	31:16	r	<b>Module Identification Number</b> 00C0 <sub>H</sub> For module identification.

**CPUx Core Identification Register****CORE\_ID****CPUx Core Identification Register****(1FE1C<sub>H</sub>)****Application Reset Value: 0000 000X<sub>H</sub>****CPU\_CORE\_ID****Short address for domain CSFR****(0FE1C<sub>H</sub>)****Application Reset Value: 0000 000X<sub>H</sub>**

Field	Bits	Type	Description
<b>CORE_ID</b>	2:0	r	<b>Core Identification Number</b> The identification number of the core.
<b>RES</b>	31:3	r	<b>Reserved</b>

## CPU Subsystem

### CPUX Customer ID register

#### CUS\_ID

CPUX Customer ID register

(1FE50<sub>H</sub>)

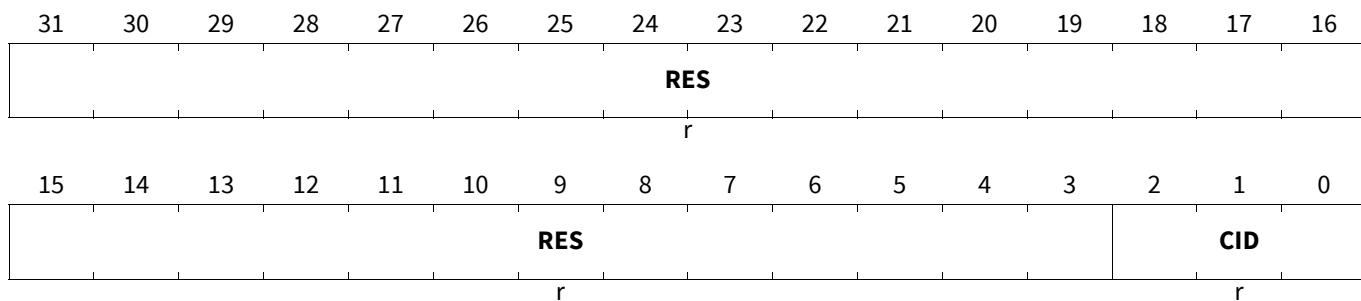
Application Reset Value: 0000 000X<sub>H</sub>

#### CPU\_CUS\_ID

Short address for domain CSFR

(0FE50<sub>H</sub>)

Application Reset Value: 0000 000X<sub>H</sub>



Field	Bits	Type	Description
CID	2:0	r	<b>Customer ID</b> See <a href="#">Chapter 5.3.6.1.4</a> for the relation between CUS_ID and CORE_ID for each derivative
RES	31:3	r	<b>Reserved</b>

### CPUX Data Access Cacheability Register

The Physical Memory Attributes registers (PMA0,PMA1,PMA2) define the physical memory attribute for each segment in the physical address space. The register is CPUX ENDINIT protected and can be read with the MFCR instruction and written by the MTCR instruction. Note that when changing the value of the registers both the instruction and data caches should be invalidated, a DSYNC instruction should be executed immediately prior to the MTCR with an ISYNC instruction executed immediately following. This is required to maintain coherency of the processors view of memory.

The register PMA0 defines the data access cacheability of the segment in the physical address space. If bit n in the register is set then segment-n will be seen as cacheable for data accesses.

PMA0 is freely programmable with the following restrictions:

- In PMA0 Segment-C and Segment[7-CORE\_ID] must have the same value.

Irrespective of the setting of the PMA registers the following constraints are always enforced.

- Segments-F and segment-E are constrained to be Peripheral space and hence non-cacheable.
- Segment-A is constrained to be non-cacheable memory
- Segment-D and Segment[7-CORE\_ID] are constrained to be non-cacheable for data accesses.

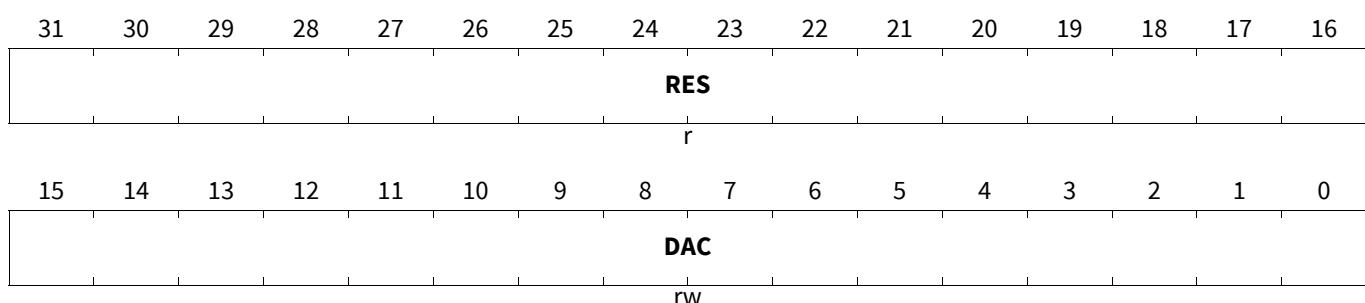
## CPU Subsystem

### PMA0

**CPUx Data Access CacheabilityRegister** **(18100<sub>H</sub>)** **Application Reset Value: 0000 0300<sub>H</sub>**

**CPU\_PMA0**

**Short address for domain CSFR** **(08100<sub>H</sub>)** **Application Reset Value: 0000 0300<sub>H</sub>**



Field	Bits	Type	Description
<b>DAC</b>	15:0	rw	<b>Data Access Cacheability Segments F<sub>H</sub>to 0<sub>H</sub></b> (Note:- segments F <sub>H</sub> , E <sub>H</sub> , D <sub>H</sub> and A <sub>H</sub> are constrained to be non-cacheable)
<b>RES</b>	31:16	r	<b>Reserved</b>

### CPUx Code Access CacheabilityRegister

The Physical Memory Attributes registers (PMA0,PMA1,PMA2) define the physical memory attribute for each segment in the physical address space. The register is CPUx ENDINIT protected and can be read with the MFCR instruction and written by the MTCR instruction. Note that when changing the value of the registers both the instruction and data caches should be invalidated, a DSYNC instruction should be executed immediately prior to the MTCR with an ISYNC instruction executed immediately following. This is required to maintain coherency of the processors view of memory.

The register PMA1 defines the code access cacheability of the segment in the physical address space. If bit n in the register is set then segment-n will be seen as cacheable for code accesses.

PMA1 is freely programmable with the following restrictions:

- In PMA1 Segment-D and Segment[7-CORE\_ID] must have the same value.

Irrespective of the setting of the PMA registers the following constraints are always enforced.

- Segments-F and segment-E are constrained to be Peripheral space and hence non-cacheable.
- Segment-A is constrained to be non-cacheable memory
- Segment-C and Segment[7-CORE\_ID] are constrained to be non-cacheable for code accesses.

## CPU Subsystem

### PMA1

**CPUx Code Access Cacheability Register**

( $18104_H$ )

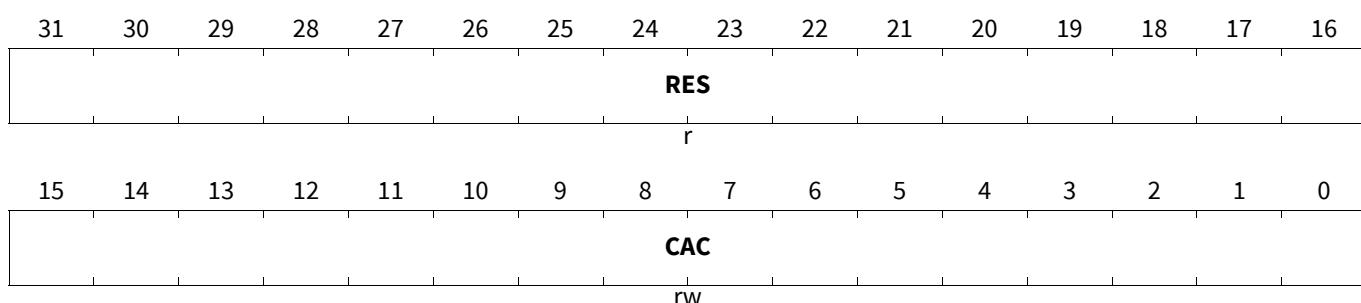
**Application Reset Value: 0000 0300<sub>H</sub>**

**CPU\_PMA1**

**Short address for domain CSFR**

( $08104_H$ )

**Application Reset Value: 0000 0300<sub>H</sub>**



Field	Bits	Type	Description
<b>CAC</b>	15:0	rw	<b>Code Access Cacheability Segments FH-0H</b> (Note: Segments F <sub>H</sub> , E <sub>H</sub> , C <sub>H</sub> , A <sub>H</sub> are constrained to be non-cacheable)
<b>RES</b>	31:16	r	<b>Reserved</b>

### CPUx Peripheral Space Identifier register

The Physical Memory Attributes registers (PMA0,PMA1,PMA2) define the physical memory attribute for each segment in the physical address space. The register is CPUx ENDINIT protected and can be read with the MFCR instruction and written by the MTCR instruction. Note that when changing the value of the registers both the instruction and data caches should be invalidated, a DSYNC instruction should be executed immediately prior to the MTCR with an ISYNC instruction executed immediately following. This is required to maintain coherency of the processors view of memory.

The register PMA2 defines the peripheral space identifier of the segment in the physical address space. If bit n in the register is set then segment-n will be seen as a peripheral segment. PMA2 is a read-only register.

### PMA2

**CPUx Peripheral Space Identifier register**

( $18108_H$ )

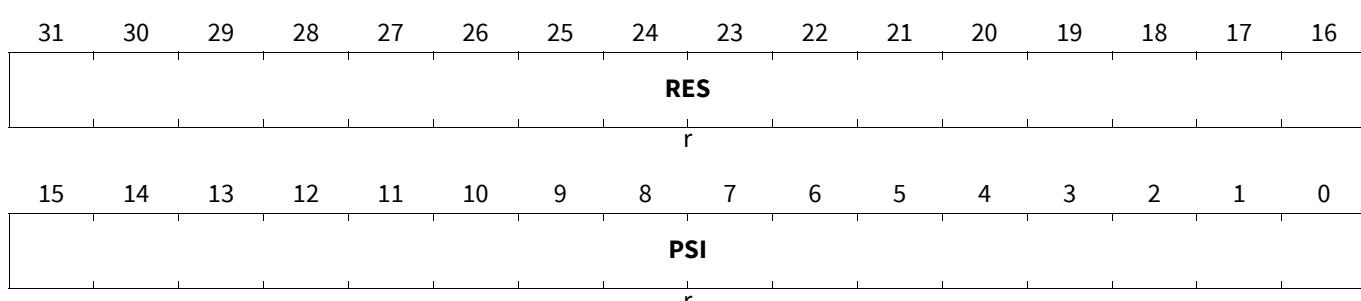
**Application Reset Value: 0000 C000<sub>H</sub>**

**CPU\_PMA2**

**Short address for domain CSFR**

( $08108_H$ )

**Application Reset Value: 0000 C000<sub>H</sub>**



Field	Bits	Type	Description
<b>PSI</b>	15:0	r	<b>Peripheral Space Identifier Segments FH-0H</b>
<b>RES</b>	31:16	r	<b>Reserved</b>

## CPU Subsystem

### CPUX Compatibility Control Register

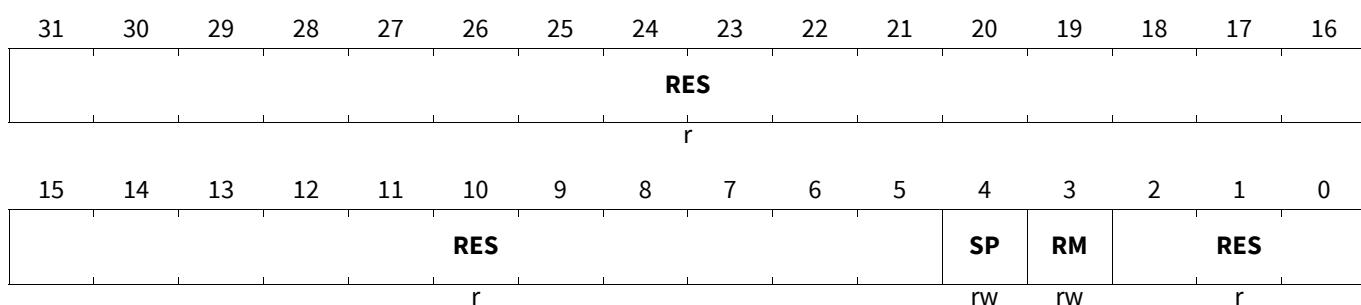
The Compatibility Control Register (COMPAT) is an implementation-specific CSFR which allows certain elements of backwards compatibility with TriCore 1.3.x behavior to be forced. The reset value of the COMPAT register ensures that backwards compatibility with TriCore 1.3 is enabled by default. This register is safety\_endinit protected

#### COMPAT

**CPUX Compatibility Control Register** **(19400<sub>H</sub>)** **Application Reset Value: FFFF FFFF<sub>H</sub>**

#### CPU\_COMPAT

**Short address for domain CSFR** **(09400<sub>H</sub>)** **Application Reset Value: FFFF FFFF<sub>H</sub>**



Field	Bits	Type	Description
<b>RES</b>	2:0, 31:5	r	<b>Reserved</b> Read as 1; should be written with 1.
<b>RM</b>	3	rw	<b>Rounding Mode Compatibility</b> 0 <sub>B</sub> PSW.RM not restored by RET. 1 <sub>B</sub> PSW.RM restored by RET (TC1.3 behavior).
<b>SP</b>	4	rw	<b>SYSCON Safety Protection Mode Compatibility</b> 0 <sub>B</sub> SYSCON[31:1] safety endinit protected. 1 <sub>B</sub> SYSCON[31:1] not safety endinit protected (TC1.3 behavior).

## CPU Subsystem

## 5.3.4.15 CPU General Purpose Registers

The only implementation specific features of the CPU General Purpose Registers are their reset value.

## 5.3.4.15.1 CPU General Purpose Registers

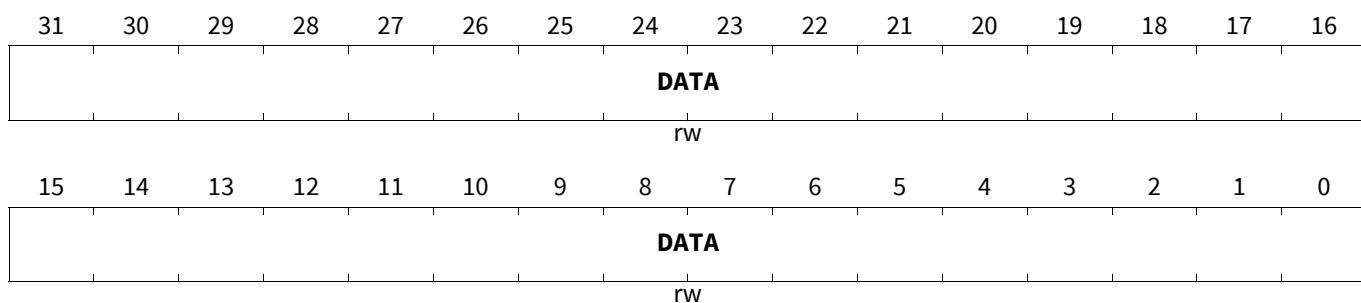
## CPUx Data General Purpose Register y

Dy (y=0-15)

CPUx Data General Purpose Register y  $(1FF00_H + y * 4)$  Application Reset Value: 0000 0000<sub>H</sub>

CPU\_Dy (y=0-15)

Short address for domain CSFR  $(0FF00_H + y * 4)$  Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
DATA	31:0	rw	<b>Data Register</b> General purpose registers

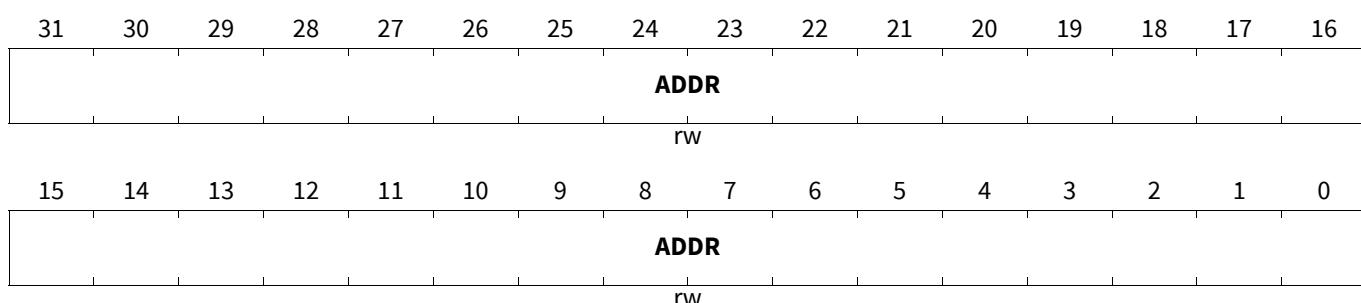
## CPUx Address General Purpose Register y

Ay (y=0-15)

CPUx Address General Purpose Register y  $(1FF80_H + y * 4)$  Application Reset Value: 0000 0000<sub>H</sub>

CPU\_Ay (y=0-15)

Short address for domain CSFR  $(0FF80_H + y * 4)$  Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
ADDR	31:0	rw	<b>Address Register</b> General purpose registers

## CPU Subsystem

## 5.3.4.16 FPU Registers

The only implementation specific features of the FPU Registers are their reset value.

## 5.3.4.16.1 FPU registers

## CPUx Trap Control Register

## FPU\_TRAP\_CON

CPUx Trap Control Register

(1A000<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

CPU\_FPU\_TRAP\_CON

Short address for domain CSFR

(0A000<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>RES</b>	<b>FI</b>	<b>FV</b>	<b>FZ</b>	<b>FU</b>	<b>FX</b>		<b>RES</b>		<b>FIE</b>	<b>FVE</b>	<b>FZE</b>	<b>FUE</b>	<b>FXE</b>		<b>RES</b>
r	rh	rh	rh	rh	rh		r		rw	rw	rw	rw	rw	r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>RES</b>						<b>RM</b>	<b>RES</b>						<b>TCL</b>	<b>TST</b>	
						rh			r				w	rh	

Field	Bits	Type	Description
<b>TST</b>	0	rh	<b>Trap Status</b> 0 <sub>B</sub> No instruction captured. The next enabled exception will cause the exceptional instruction to be captured. 1 <sub>B</sub> Instruction captured. No further enabled exceptions will be captured until TST is cleared.
<b>TCL</b>	1	w	<b>Trap Clear</b> Read: always reads as 0. 0 <sub>B</sub> No effect. 1 <sub>B</sub> Clears the trapped instruction (TST will be negated).
<b>RES</b>	7:2, 17:10, 25:23, 31	r	<b>Reserved</b> <b>Read as 0; should be written with 0.</b>
<b>RM</b>	9:8	rh	<b>Captured Rounding Mode</b> The rounding mode of the captured instruction. Only valid when TST is asserted. Note that this is the rounding mode supplied to the FPU for the exceptional instruction. UPDFL instructions may cause a trap and change the rounding mode. In this case the RM bits capture the input rounding mode
<b>FXE</b>	18	rw	<b>FX Trap Enable</b> When set, an instruction generating an FX exception will trigger a trap.
<b>FUE</b>	19	rw	<b>FU Trap Enable</b> When set, an instruction generating an FU exception will trigger a trap.

## CPU Subsystem

Field	Bits	Type	Description
<b>FZE</b>	20	rw	<b>FZ Trap Enable</b> When set, an instruction generating an FZ exception will trigger a trap.
<b>FVE</b>	21	rw	<b>FV Trap Enable</b> When set, an instruction generating an FV exception will trigger a trap.
<b>FIE</b>	22	rw	<b>FI Trap Enable</b> When set, an instruction generating an FI exception will trigger a trap.
<b>FX</b>	26	rh	<b>Captured FX</b> Asserted if the captured instruction asserted FX. Only valid when TST is asserted.
<b>FU</b>	27	rh	<b>Captured FU</b> Asserted if the captured instruction asserted FU. Only valid when TST is asserted.
<b>FZ</b>	28	rh	<b>Captured FZ</b> Asserted if the captured instruction asserted FZ. Only valid when TST is asserted
<b>FV</b>	29	rh	<b>Captured FV</b> Asserted if the captured instruction asserted FV. Only valid when TST is asserted
<b>FI</b>	30	rh	<b>Captured FI</b> Asserted if the captured instruction asserted FI. Only valid when TST is asserted

## CPUx Trapping Instruction Program Counter Register

### FPU\_TRAP\_PC

CPUx Trapping Instruction Program Counter Register( $1A004_H$ )

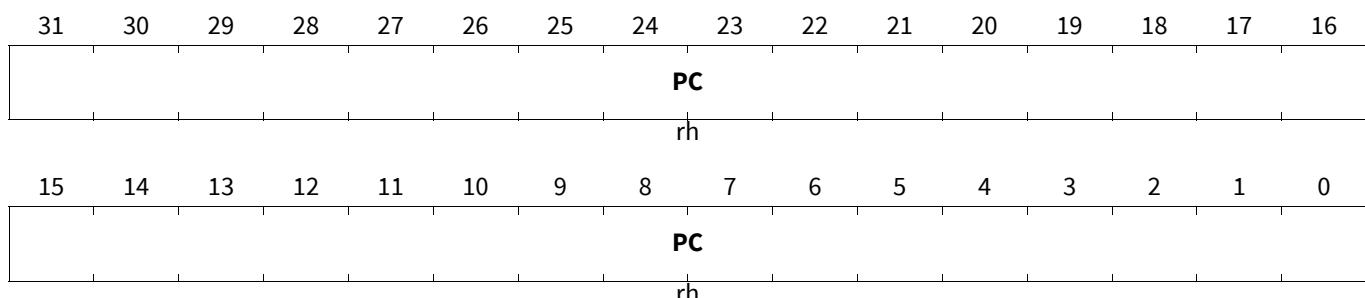
Application Reset Value:  $0000\ 0000_H$

### CPU\_FPU\_TRAP\_PC

Short address for domain CSFR

( $0A004_H$ )

Application Reset Value:  $0000\ 0000_H$



Field	Bits	Type	Description
<b>PC</b>	31:0	rh	<b>Captured Program Counter</b> The program counter (virtual address) of the captured instruction. Only valid when FPU_TRAP_CON.TST is asserted.

## CPU Subsystem

### CPUx Trapping Instruction Opcode Register

#### FPU\_TRAP\_OPCODE

CPUx Trapping Instruction Opcode Register **(1A008<sub>H</sub>)**

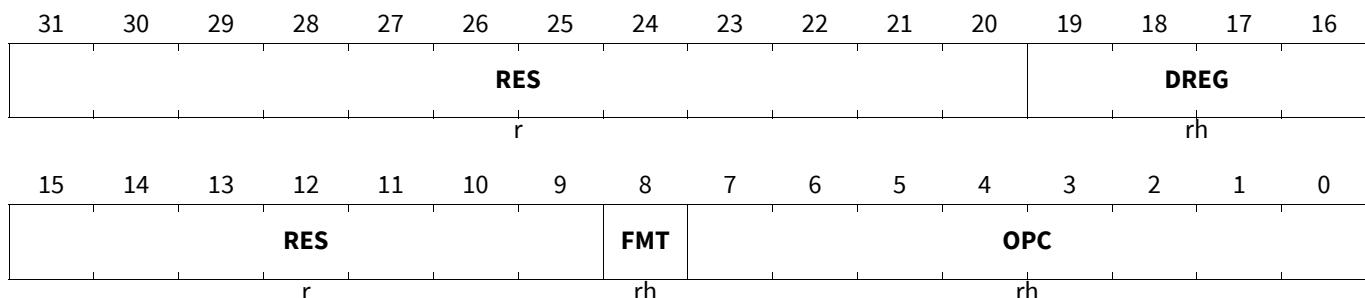
Application Reset Value: 0000 0000<sub>H</sub>

#### CPU\_FPU\_TRAP\_OPCODE

Short address for domain CSFR

**(0A008<sub>H</sub>)**

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>OPC</b>	7:0	rh	<b>Captured Opcode</b> The secondary opcode of the captured instruction. When FPU_TRAP_OPCODE.FMT=0 only bits [3:0] are defined. OPC is valid only when FPU_TRAP_CON.TST is asserted.
<b>FMT</b>	8	rh	<b>Captured Instruction Format</b> The format of the captured instruction's opcode. Only valid when FPU_TRAP_CON.TST is asserted. $0_B$ RRR $1_B$ RR
<b>RES</b>	15:9, 31:20	r	<b>Reserved</b> <b>Read as 0; should be written with 0.</b>
<b>DREG</b>	19:16	rh	<b>Captured Destination Register</b> The destination register of the captured instruction. ... Only valid when FPU_TRAP_CON.TST is asserted. $0_H$ Data general purpose register 0. $F_H$ Data general purpose register 15.

## CPU Subsystem

### CPUX Trapping Instruction Operand Register

#### FPU\_TRAP\_SRC1

CPUX Trapping Instruction Operand Register (1A010<sub>H</sub>)

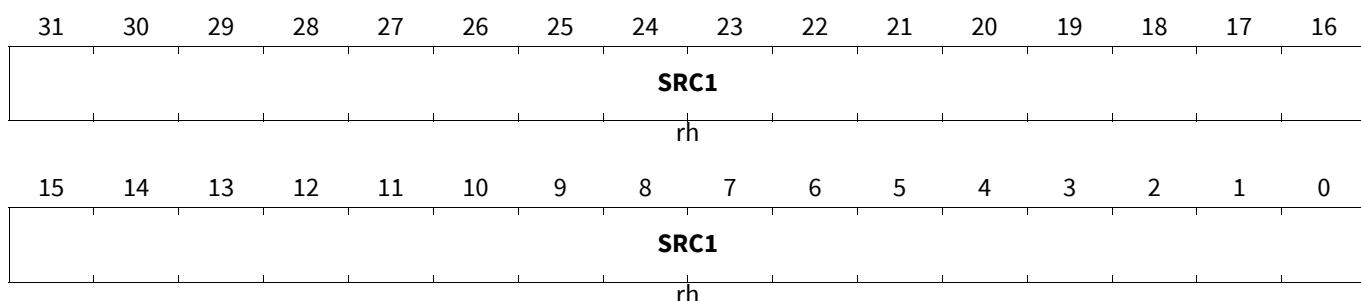
Application Reset Value: 0000 0000<sub>H</sub>

#### CPU\_FPU\_TRAP\_SRC1

Short address for domain CSFR

(0A010<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
SRC1	31:0	rh	<b>Captured SRC1 Operand</b> The SRC1 operand of the captured instruction. Only valid when FPU_TRAP_CON.TST is asserted.

### CPUX Trapping Instruction Operand Register

#### FPU\_TRAP\_SRC2

CPUX Trapping Instruction Operand Register (1A014<sub>H</sub>)

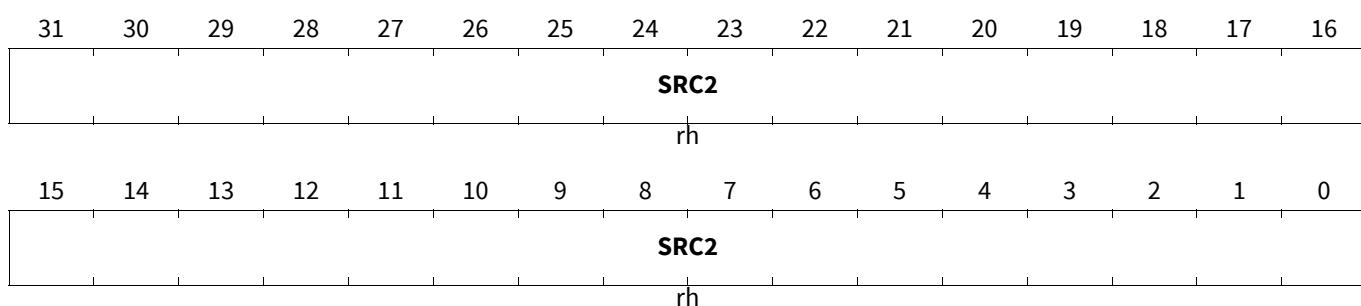
Application Reset Value: 0000 0000<sub>H</sub>

#### CPU\_FPU\_TRAP\_SRC2

Short address for domain CSFR

(0A014<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
SRC2	31:0	rh	<b>Captured SRC2 Operand</b> The SRC2 operand of the captured instruction. Only valid when FPU_TRAP_CON.TST is asserted.

## CPU Subsystem

### CPUx Trapping Instruction Operand Register

#### FPU\_TRAP\_SRC3

CPUx Trapping Instruction Operand Register (1A018<sub>H</sub>)

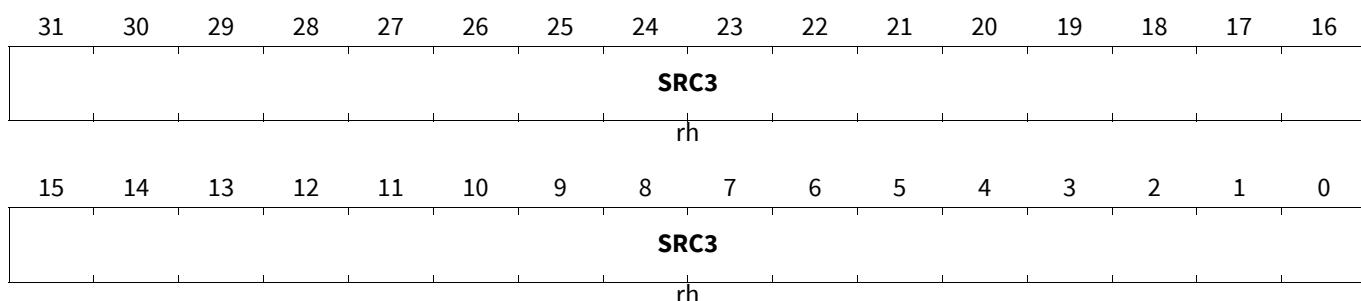
Application Reset Value: 0000 0000<sub>H</sub>

#### CPU\_FPU\_TRAP\_SRC3

Short address for domain CSFR

(0A018<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
SRC3	31:0	rh	<b>Captured SRC3 Operand</b> The SRC3 operand of the captured instruction. Only valid when FPU_TRAP_CON.TST is asserted.

#### 5.3.4.17 CPU Memory Protection Registers

There are six Memory Protection Register Sets per CPU. The sets specify memory protection ranges and permissions for code and data. The PSW.PRS bit field determines which of these sets is currently in use by the CPU. (Note: the three bit PRS field is composed as PRS[2:0] == {PSW[15],PSW[13],PSW[12]})

The CPUs each implement 18 data and 10 code range comparators. These may be flexibly shared amongst the protection sets to provide a maximum of 18 data ranges and 10 code ranges per set.

The protection registers protect the whole address space (In previous TriCore implementations the memory protection system did not cover peripheral space.) The granularity of the protection ranges is 8 bytes for data and 32bytes for code.

Code protection ranges define which memory areas the CPU may fetch instructions from. Instruction fetches from an area outside a valid code protection range will lead to a trap condition.

Data protection ranges define which memory areas the CPU may access for read and/or write operations. Each range may be separately enabled for read and/or write access. Data read accesses from an area outside a valid data protection range with read permissions will lead to a trap condition. Data write accesses to an area outside a valid data protection range with write permissions will lead to a trap condition.

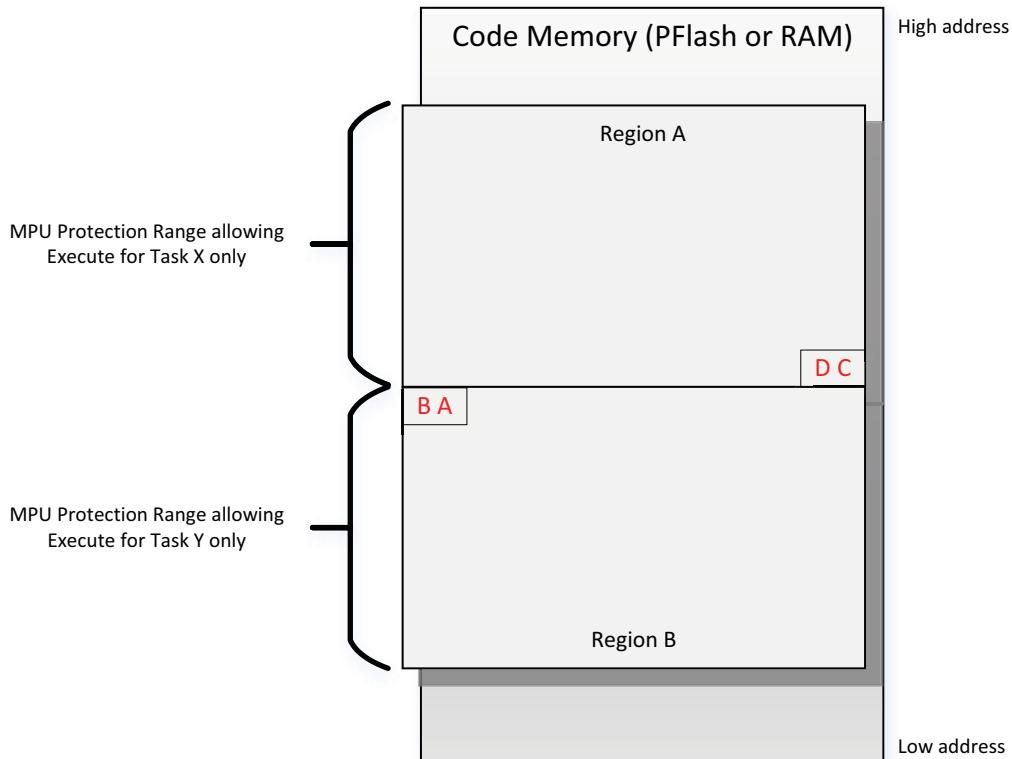
There are no implementation specific features of the Memory Protection Registers. They are described in detail in the TriCore Architecture Manual.

#### Crossing protection boundaries

When protection is enabled the TC1.6.2P checks the base address of instruction fetch and data access operations against the enabled MPU protection ranges. If this base address is outside of the enabled protection ranges a trap will be raised. The TC1.6.2P does not check the full extent of instruction fetch or data access. An instruction fetch or data access that starts at the very top of an enabled region may therefore extend into a non-enabled region without triggering a trap condition.

## CPU Subsystem

In the following example a 4-byte instruction “DCBA” is placed at valid location “A” in protection Region-B but extends into invalid Region-A. Execution of this instruction will not lead to an MPX trap condition even though a portion of the instruction extends into an invalid region.



**Figure 50 Crossing protection boundaries**

To maintain separation between protection regions the following buffers regions should be placed between each memory protection region.

Code protection regions

- Buffer size 2 Bytes<sup>1)\*</sup>

Data protection regions containing STLCX or STUCX context data

- Buffer size 56 Bytes

Data protection regions not containing STLCX or STUCX context data

- Buffer size 6 bytes\*

1) In reality the minimum buffer size will be 8-bytes due to protection region granularity.

## CPU Subsystem

**5.3.4.18 Temporal Protection Registers**

To guard against task runtime overrun the CPU implements a temporal protection system. This system consists of three independent decrementing counters (TPS\_TIMERn) arranged to generate a Temporal Asynchronous Error trap (TAE - Class-4, Tin-7) on decrement to zero. A control register (TPS\_CON) contains status and control bits for temporal protection system. The temporal protection system is enabled by the SYSCON.TPROTEN register bit. The Temporal Protection Registers are Core Special Function Registers. They are described in detail in the TriCore Architecture Manual.

While the CPU is in Suspen the temporal protection timer clocks are frozen. The Timers will restart once the CPU leaves Suspend mode.

**5.3.4.18.1 Temporal Protection Registers****CPUx Temporal Protection System Control Register**

Definition of the Temporal Protection System Control register.

**TPS\_CON**

**CPUx Temporal Protection System Control Register(1E400<sub>H</sub>)**

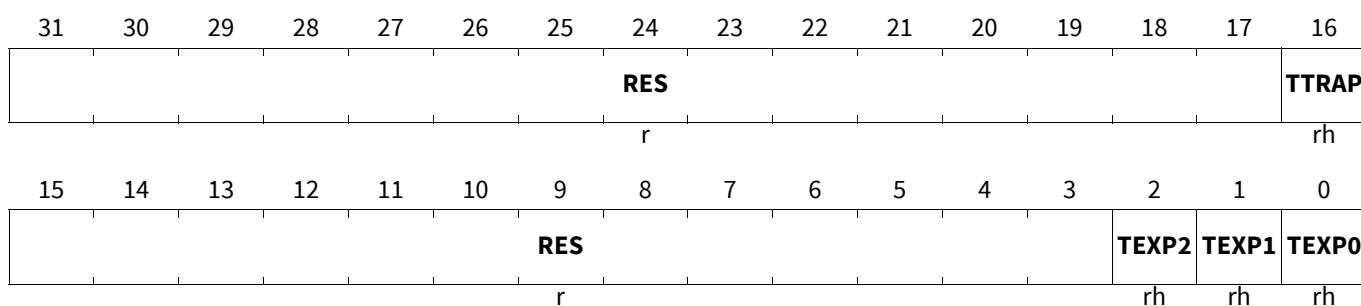
**Application Reset Value: 0000 0000<sub>H</sub>**

**CPU\_TPS\_CON**

**Short address for domain CSFR**

**(0E400<sub>H</sub>)**

**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>TEXPO</b>	0	rh	<b>Timer0 Expired Flag</b> Set when the corresponding timer expires. Cleared on any write to the _TIMER0 register.
<b>TEXP1</b>	1	rh	<b>Timer1 Expired Flag</b> Set when the corresponding timer expires. Cleared on any write to the _TIMER1 register.
<b>TEXP2</b>	2	rh	<b>Timer1 Expired Flag</b> Set when the corresponding timer expires. Cleared on any write to the _TIMER1 register.
<b>RES</b>	15:3, 31:17	r	<b>Reserved</b>
<b>TTRAP</b>	16	rh	<b>Temporal Protection Trap</b> If set, indicates that a TAE trap has been requested. Any subsequent TAE traps are disabled. A write clears the flag and re-enables TAE traps.

## CPU Subsystem

### CPUX Temporal Protection System Timer Register y

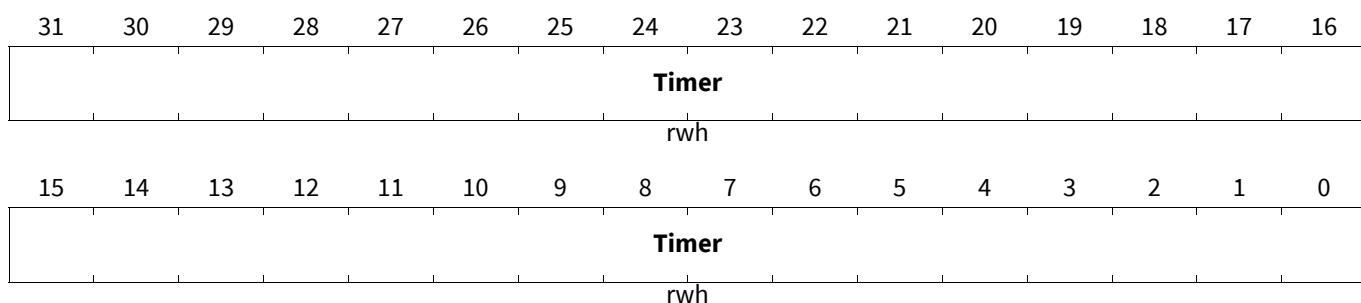
Definition of the Temporal Protection System Timer register y.

#### TPS\_TIMERy (y=0-2)

**CPUX Temporal Protection System Timer Register y(1E404<sub>H</sub>+y\*4) Application Reset Value: 0000 0000<sub>H</sub>**

#### CPU\_TPS\_TIMERy (y=0-2)

**Short address for domain CSFR (0E404<sub>H</sub>+y\*4) Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
Timer	31:0	rwh	<b>Temporal Protection Timer</b> Writing zero de-activates the Timer. Writing a non-zero value starts the Timer. Any write clears the corresponding TPS_CON.TEXP flag. Read returns the current Timer value.

#### 5.3.4.19 Exception Timer

For AURIX™ the temporal protection system is extended to include a dedicated exception timer. If a selected Trap or NMI is not serviced within a defined time period a safety alarm is generated. The exception timer is part of the TriCore temporal protection system and must be enabled by setting the SYSCON.TPROTEN bit.

There are two distinct periods during the handling of an exception. The first period is between the CPU receiving a trap request and the start of the entry sequence for that particular trap. The second period is between trap handler entry and the exit of the trap handler with an RFE instruction. These two periods are separately timed with an 12-bit exception entry timer and a 24-bit exception exit timer respectively.

The exception entry timer is loaded with a predefined value on receipt of an enabled class of NMI or trap, it then decrements each cycle until the entry sequence for the trap is detected. If the timer decrements to zero a safety alarm is raised to be handled by the SMU.

The detailed function of the exception entry timer is as follows:-

- The exception entry timer load value is held in the register TPS\_EXTIM\_ENTRY\_LVAL.ENTRY\_LVAL.
- If the ENTRY\_LVAL value is zero the exception entry timer is disabled.
- The four least significant bits of ENTRY\_LVAL are constrained to be zero.
- The exception entry timer current value is held in the register TPS\_EXTIM\_ENTRY\_CVAL.ENTRY\_CVAL
- When inactive the exception entry timer has a ENTRY\_CVAL value of 0.
- If inactive then on receipt of an exception entry timer start request the exception entry timer will be loaded with the ENTRY\_LVAL value and start decrementing. If the exception entry timer has a non-zero value when a start request is received then no action is taken.

## CPU Subsystem

- If the exception entry timer is inactive then on an exception entry timer start request the class and tin of the requesting trap are recorded in the TPS\_EXTIM\_STAT.ENTRY\_CLASS and TPS\_EXTIM\_STAT.ENTRY\_TIN registers.
- The exception entry timer will decrement by 1 on each processor clock when non-zero. If it decrements from 1 to 0 a timeout alarm is generated to the SMU.
- An exception entry timer start request is generated on receipt of a trap or NMI request for an enabled trap class.
- The register TPS\_EXTIM\_CLASS\_EN contains an 8 bit field with a separate enable for each of the defined trap classes. Only classes of trap for which the enable bit is set will generate an exception entry timer start request.
- The exception entry timer stops decrementing when an entry sequence commences for the trap which generated the start request. The TPS\_EXTIM\_ENTRY\_CVAL.EXIT\_CVAL, TPS\_EXTIM\_ENTRY\_STAT.ENTRY\_CLASS and TPS\_EXTIM\_ENTRY\_STAT.ENTRY\_TIN are cleared
- In the AURIX™ TriCore implementation synchronous traps are always taken as soon as they are requested hence only asynchronous traps and NMIs are timed by the exception entry timer.
- When the exception entry timer is disabled, TPS\_EXTIM\_STAT.ENTRY\_CLASS and TPS\_EXTIM\_STAT.ENTRY\_TIN are set to 0 unless either \_AT bits of the TPS\_EXTIM\_STAT register is set

The exception exit timer is loaded with a predefined value on detection of an entry sequence for an enabled class of trap or NMI, then decrements each cycle until the RFE indicating the exit of the trap or NMI handler is detected. If the timer decrements to zero a safety alarm is raised to be handled by the SMU.

The detailed function of the exception exit timer is as follows:-

- The exception exit timer load value is held in the register TPS\_EXTIM\_EXIT\_LVAL.EXIT\_LVAL
- If the EXIT\_LVAL value is zero the exception exit timer is disabled.
- The four least significant bits of EXIT\_LVAL are constrained to be zero.
- The exception exit timer current value is held in the register TPS\_EXTIM\_EXIT\_CVAL.EXIT\_CVAL
- When inactive the exception exit timer has a EXIT\_CVAL value of 0.
- If inactive then on receipt of an exception exit timer start request the exception exit timer will be loaded with the EXIT\_LVAL value and start decrementing. If the exception exit timer has a non-zero value when a start request is received then no action is taken.
- If the exception exit timer is inactive then on an exception exit start request the current value of FCX is recorded in the TPS\_EXTIM\_STAT.EXIT\_FCX register and the class and tin of the trap are recorded in the TPS\_EXTIM\_STAT.EXIT\_CLASS and TPS\_EXTIM\_STAT.EXIT\_TIN registers.
- The exception exit timer will decrement by 1 on each processor clock when non-zero. If it decrements from 1 to 0 a timeout alarm is generated to the SMU.
- An exception exit timer start request is generated on entry to a trap or NMI handler for an enabled trap class.
- The register TPS\_EXTIM\_CLASS\_EN contains an 8bit field with a separate enable for each of the defined trap or NMI classes. Only classes of trap for which the enable bit is set will generate an exception exit timer start request.
- The exception exit timer stops decrementing when an RFE is executed for the trap which generated the start request. The TPS\_EXTIM\_EXIT\_CVAL.EXIT\_CVAL, TPS\_EXTIM\_FCX.EXIT\_FCX, TPS\_EXTIM\_STAT.EXIT\_CLASS and TPS\_EXTIM\_STAT.EXIT\_TIN are cleared
- To ensure the exit timer is only stopped for the correct trap the current FCX value at exit timer start is stored into the TPS\_EXTIM\_FCX.EXIT\_FCX register. On execution of an RFE instruction the return FCX value (the value of FCX after the RFE) is compared with the FCX value held in the TPS\_EXTIM\_FCX.EXIT\_FCX register. If they match then this indicates the successful completion of the trap handler and the exception exit timer is cleared to zero.

## CPU Subsystem

- When the exception exit timer is disabled, TPS\_EXTIM\_STAT.EXIT\_CLASS, TPS\_EXTIM\_STAT.EXIT\_TIN and TPS\_EXTIM\_FCX are set to 0 unless either \_AT bits of the TPS\_EXTIM\_STAT register is set.

The current status of the exception timer is held in the TPS\_EXTIM\_STAT register:-

- If the exception entry timer times out the TPS\_EXTIM\_STAT.ENTRY\_AT bit is set and an alarm raised to the SMU
- If the exception exit timer times out the TPS\_EXTIM\_STAT.EXIT\_AT bit is set and an alarm raised to the SMU
- If either of the \_AT bits is set then the current state of the TPS\_EXTIM\_ENTRY\_CVAL, TPS\_EXTIM\_EXIT\_CVAL, TPS\_EXTIM\_FCX and TPS\_EXTIM\_STAT register are held until the TPS\_EXTIM\_STAT register is cleared.
- The exception timer may be stopped and cleared by writing the reverse of the current TPS\_EXTIM\_STAT contents to the TPS\_EXTIM\_STAT register location (all other values will be ignored). This allows trap handlers to safely modify the context save areas where required without risk of reset. The TPS\_EXTIM\_ENTRY\_CVAL, TPS\_EXTIM\_EXIT\_CVAL, EXTIM\_EXIT\_FCX and TPS\_EXTIM\_STAT registers are cleared by this operation.
- If an enabled asynchronous trap is pending (i.e. the entry timer is active) then attempting to stop the exception timer by writing the TPS\_EXTIM\_STAT register will not result in the entry timer being reset.
- The exception timer system is clocked whenever the primary clock input to the core is active (it is therefore active during idle).

## CPU Subsystem

### 5.3.4.19.1 Exception Timers Registers

#### CPUx Exception Entry Timer Load Value

This register contains the count value to be loaded into the exception timer on detection of an enabled exception.

##### TPS\_EXTIM\_ENTRY\_LVAL

CPUx Exception Entry Timer Load Value **(1E440<sub>H</sub>)**

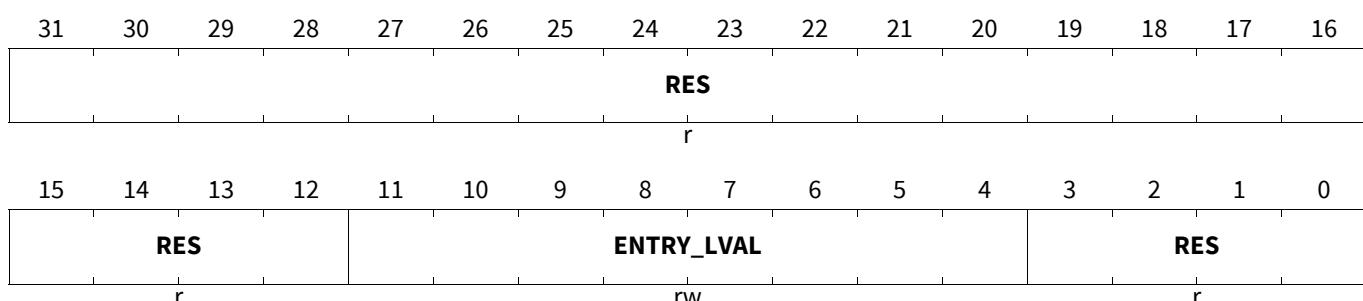
**Application Reset Value: 0000 0000<sub>H</sub>**

##### CPU\_TPS\_EXTIM\_ENTRY\_LVAL

Short address for domain CSFR

**(0E440<sub>H</sub>)**

**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>RES</b>	3:0, 31:12	r	<b>Reserved</b> The lower 4 bits are not writeable and always return zero
<b>ENTRY_LVAL</b>	11:4	rw	<b>Exception Entry Timer Load value</b> Value loaded into the exception entry timer on detection of an enabled exception. Bits [3:0] are constrained to be 0

#### CPUx Exception Exit Timer Load Value

This register contains the count value to be loaded into the exception timer on detection of an enabled exception.

##### TPS\_EXTIM\_EXIT\_LVAL

CPUx Exception Exit Timer Load Value **(1E448<sub>H</sub>)**

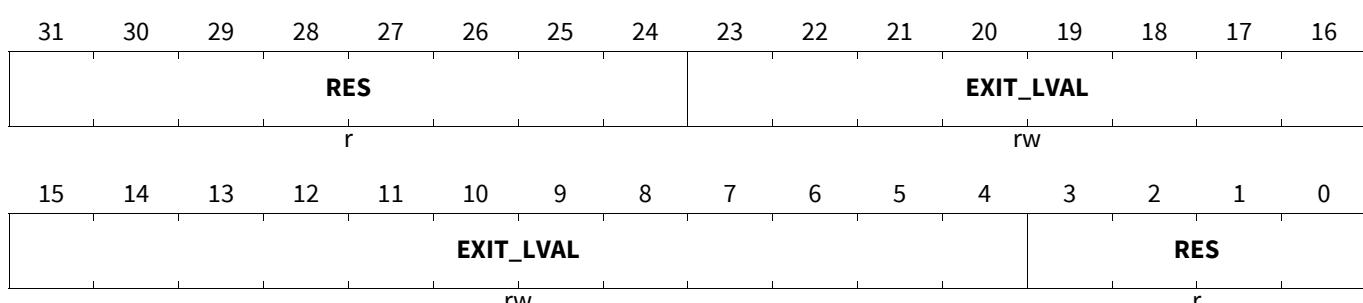
**Application Reset Value: 0000 0000<sub>H</sub>**

##### CPU\_TPS\_EXTIM\_EXIT\_LVAL

Short address for domain CSFR

**(0E448<sub>H</sub>)**

**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>RES</b>	3:0, 31:24	r	<b>Reserved</b> The lower 4 bits are not writeable and always return zero

**CPU Subsystem**

Field	Bits	Type	Description
<b>EXIT_LVAL</b>	23:4	rw	<b>Exception Exit Timer Load value</b> Value loaded into the exception exit timer on detection of an enabled exception. Bits [3:0] are constrained to be 0

**CPUx Exception Entry Timer Current Value**

This register contains the current count value of the exception entry timer. A non-zero value indicates that the counter is active.

**TPS\_EXTIM\_ENTRY\_CVAL**

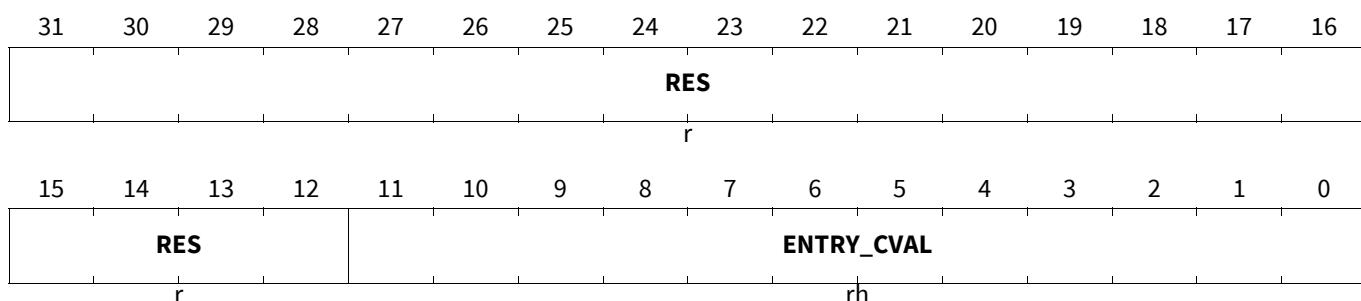
**CPUx Exception Entry Timer Current Value** **(1E444H)**

**Application Reset Value: 0000 0000H**

**CPU\_TPS\_EXTIM\_ENTRY\_CVAL**

**Short address for domain CSFR** **(0E444H)**

**Application Reset Value: 0000 0000H**



Field	Bits	Type	Description
<b>ENTRY_CVAL</b>	11:0	rh	<b>Exception Entry Timer Current Value</b> Current value of the exception entry timer.
<b>RES</b>	31:12	r	<b>Reserved</b>

**CPUx Exception Exit Timer Current Value**

This register contains the current count value of the exception exit timer. A non-zero value indicates that the counter is active.

**TPS\_EXTIM\_EXIT\_CVAL**

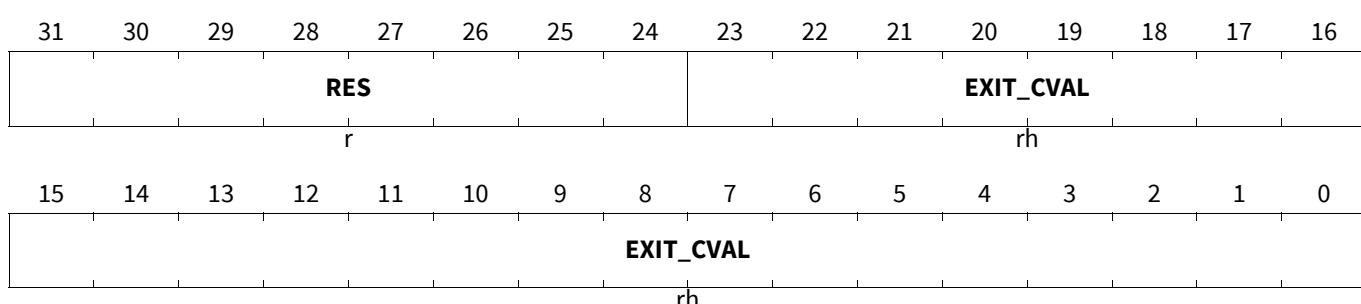
**CPUx Exception Exit Timer Current Value** **(1E44CH)**

**Application Reset Value: 0000 0000H**

**CPU\_TPS\_EXTIM\_EXIT\_CVAL**

**Short address for domain CSFR** **(0E44CH)**

**Application Reset Value: 0000 0000H**



## CPU Subsystem

Field	Bits	Type	Description
<b>EXIT_CVAL</b>	23:0	rh	<b>Exception Exit Timer Current Value</b> Current value of the exception exit timer.
<b>RES</b>	31:24	r	<b>Reserved</b>

### CPUx Exception Timer Class Enable Register

This register contains the class enables for the exception timer. Only those traps with an enabled class will trigger an exception timer start.

#### TPS\_EXTIM\_CLASS\_EN

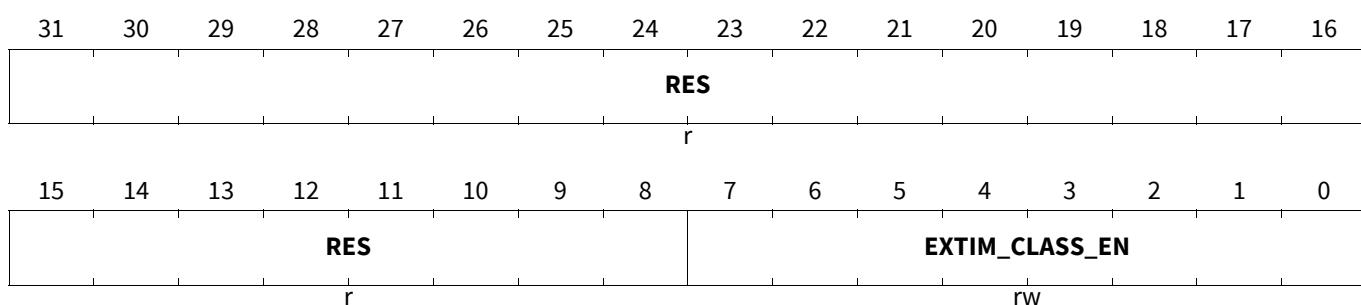
**CPUx Exception Timer Class Enable Register**  $(1E450_H)$

**Application Reset Value:** 0000 0000<sub>H</sub>

#### CPU\_TPS\_EXTIM\_CLASS\_EN

**Short address for domain CSFR**  $(0E450_H)$

**Application Reset Value:** 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>EXTIM_CLASS_EN</b>	7:0	rw	<b>Exception Timer Class Enables</b> Trap Class enables for exception timer.
<b>RES</b>	31:8	r	<b>Reserved</b>

### CPUx Exception Timer Status Register

This register contains information of the last trap to trigger the exception timer. A write of the reverse of the current value to this register will stop the exception timer and clear this register and the TPS\_EXTIM\_CVAL and TPS\_EXTIM\_FCX register.

#### TPS\_EXTIM\_STAT

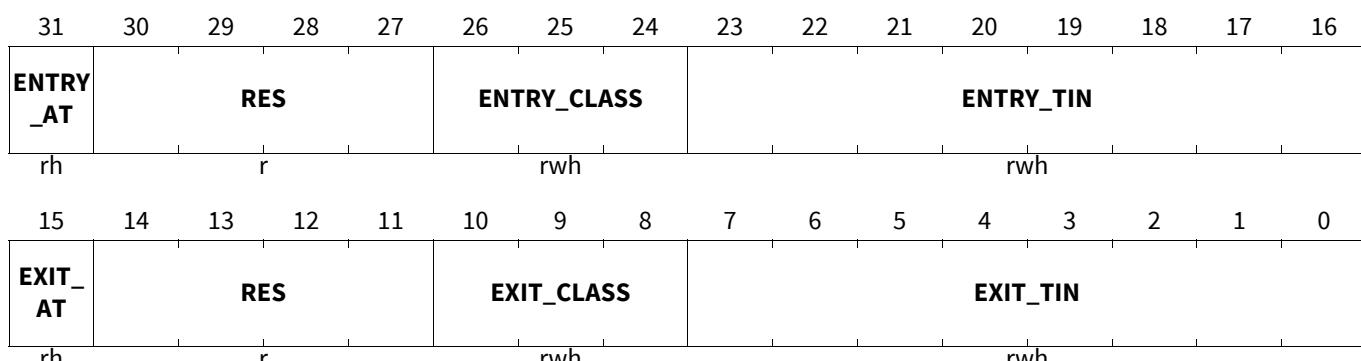
**CPUx Exception Timer Status Register**  $(1E454_H)$

**Application Reset Value:** 0000 0000<sub>H</sub>

#### CPU\_TPS\_EXTIM\_STAT

**Short address for domain CSFR**  $(0E454_H)$

**Application Reset Value:** 0000 0000<sub>H</sub>



## CPU Subsystem

Field	Bits	Type	Description
<b>EXIT_TIN</b>	7:0	rwh	<b>Exception Exit Timer TIN</b> Exception Exit Timer TIN of triggering trap.
<b>EXIT_CLASS</b>	10:8	rwh	<b>Exception Exit Timer Class</b> Exception exit Timer Class of triggering trap.
<b>RES</b>	14:11, 30:27	r	<b>Reserved</b>
<b>EXIT_AT</b>	15	rh	<b>Exception Exit Timer Alarm Triggered</b> Exception Exit Timer Alarm triggered sticky bit. Alarm triggered since last cleared.
<b>ENTRY_TIN</b>	23:16	rwh	<b>Exception Entry Timer TIN</b> Exception Entry Timer TIN of triggering trap.
<b>ENTRY_CLASS</b>	26:24	rwh	<b>Exception Entry Timer Class</b> Exception Entry Timer Class of triggering trap.
<b>ENTRY_AT</b>	31	rh	<b>Exception Entry Timer Alarm Triggered</b> Exception Entry Timer Alarm triggered sticky bit. Alarm triggered since last cleared.

### CPUx Exception Timer FCX Register

This register contains FCX information of the last trap to trigger the exception exit timer.

#### TPS\_EXTIM\_FCX

#### CPUx Exception Timer FCX Register

(1E458<sub>H</sub>)

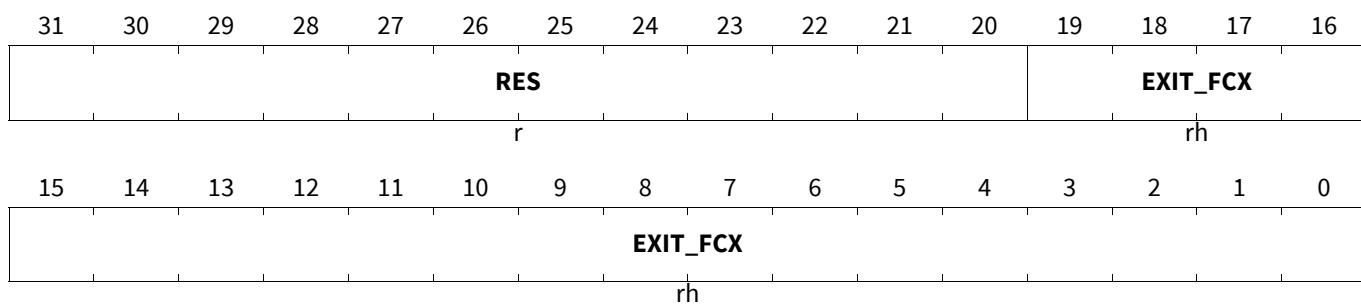
Application Reset Value: 0000 0000<sub>H</sub>

#### CPU\_TPS\_EXTIM\_FCX

#### Short address for domain CSFR

(0E458<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>EXIT_FCX</b>	19:0	rh	<b>Exception Exit Timer FCX</b> Exception Exit Timer FCX of triggering trap.
<b>RES</b>	31:20	r	<b>Reserved</b>

## CPU Subsystem

### 5.3.4.20 Memory Integrity Registers

To monitor and debug the integrity of the memory subsystems the following registers are provided.

#### 5.3.4.20.1 Register Descriptions

##### Program Integrity Error Information Registers

Two architecturally visible registers (PIETR, PIEAR) allow software to localise the source of the last detected program memory integrity error.

These registers are updated when a program integrity error condition is detected and the PIETR.IED bit is zero. On update the PIETR.IED bit is set to one and remains set until cleared by software. Whilst PIETR.IED is set further hardware updates of PIETR and PIEAR are inhibited. The various error scenarios are defined below.

##### Program Integrity errors during instruction fetch from program memory

When an error is detected during a program fetch from a program memory the IE\_S, IE\_C, IE\_T, IE\_LPB bits are updated to denote in which memory structure the error was detected. If the IE\_C bit is set the E\_INFO field will be updated with the cache way. The PIEAR register is updated with the address of the access. If the error detected is an uncorrectable bit error the IE\_UNC bit is set, The IED bit is set and all other PIETR register bits are set to zero. Since instruction fetches are speculative, the PIETR and PIEAR registers may be updated without a corresponding PIE trap

##### Program Integrity errors during a memory read initiated by an external access

When an error is detected during an external bus read from program memory the IE\_S, IE\_C bits are updated to denote in which memory structure the error was detected. The IE\_BS is set and E\_INFO updated. The PIEAR register is updated with the address of the access. If the error detected is an uncorrectable error the IE\_UNC bit is set, The IED bit is set and all other PIETR register bits are set to zero. Note that a memory read can be generated by a sub word write operation. The IE\_C bit can only ever be set if the cache is mapped into memory in SIST mode. In this case the E\_INFO field indicates the tag of the requesting master.

##### Program Integrity errors due to safety protection

When a safety protection violation is detected during a program memory access or during access to the CPU registers the IE\_SP, IE\_BS bits are set and the E\_INFO field updated. The PIEAR register is updated with the address of the access. The IED bit is set and all other PIETR register bits are set to zero.

##### Program Integrity errors due to ECC errors at the bus interface

When an ECC error is detected at the program bus interface (either master or slave) the IE\_BI bit is set. If the error is during an external access the IE\_BS bit is set and the E\_INFO fields updated. The PIEAR register is updated with the address of the access. If the error detected is an uncorrectable error the IE\_UNC bit is set. The IED bit is set and all other PIETR register bits are set to zero. If an ECC error is detected in the address phase of a request at the bus slave interface Then the IE\_ADDR and IED bits are set, all other bits are cleared.

##### Program integrity error - Memory Test Mode Violation

The PIETR.IE\_MTMV bit is set if the cache controller attempt to access the PMem (PCache/PSPR) or Ptag memories while they are in a test mode.

## CPU Subsystem

### CPUx Program Integrity Error Trap Register

#### PIETR

CPUx Program Integrity Error Trap Register (19214<sub>H</sub>)

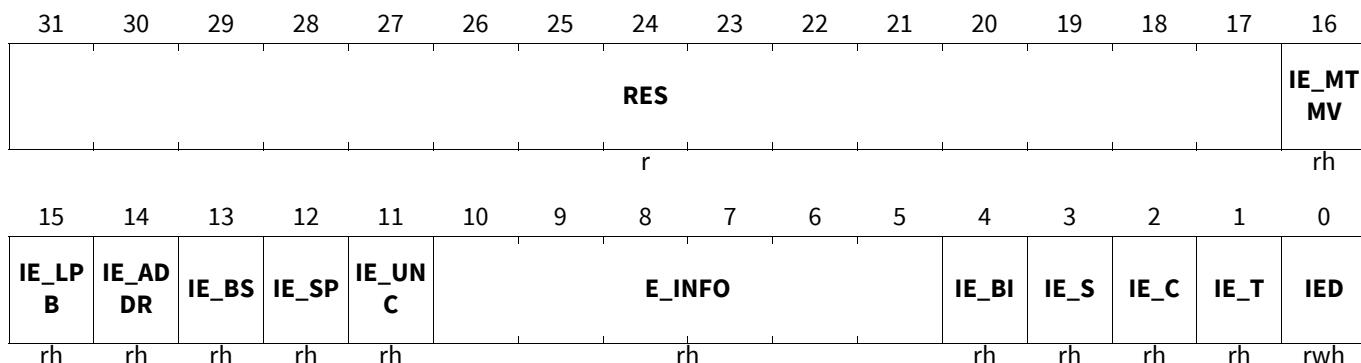
Application Reset Value: 0000 0000<sub>H</sub>

#### CPU\_PIETR

Short address for domain CSFR

(09214<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>IED</b>	0	rwh	<b>Integrity Error Detected</b> 0 <sub>B</sub> Write: Clear IED bit, re-enable PIETR and PIEAR update. Read : No data integrity error condition occurred 1 <sub>B</sub> Write : No Effect. Read: Data integrity error condition detected. PIETR and PIEAR contents valid, further PIETR and PIEAR updates disabled..
<b>IE_T</b>	1	rh	<b>Integrity Error - TAG Memory</b>
<b>IE_C</b>	2	rh	<b>Integrity Error - Cache Memory</b>
<b>IE_S</b>	3	rh	<b>Integrity Error - Scratchpad Memory</b>
<b>IE_BI</b>	4	rh	<b>Integrity Error - Bus Interface</b>
<b>E_INFO</b>	10:5	rh	<b>Error Information</b> If IE_BS= 1: Bus Master Tag ID of requesting masterIf IE_C = 1: Cache way.
<b>IE_UNC</b>	11	rh	<b>Integrity Error - Uncorrectable Error Detected</b>
<b>IE_SP</b>	12	rh	<b>Safety Protection Error Detected</b>
<b>IE_BS</b>	13	rh	<b>Bus Slave Access Indicator</b>
<b>IE_ADDR</b>	14	rh	<b>Address Phase error detected at SRI slave interface</b>
<b>IE_LPB</b>	15	rh	<b>Integrity Error - Local Pflash bank</b>
<b>IE_MTMV</b>	16	rh	<b>Memory Test Mode Violation detected</b>
<b>RES</b>	31:17	r	<b>Reserved</b>

## CPU Subsystem

### CPUx Program Integrity Error Address Register

#### PIEAR

CPUx Program Integrity Error Address Register( $19210_{\text{H}}$ )

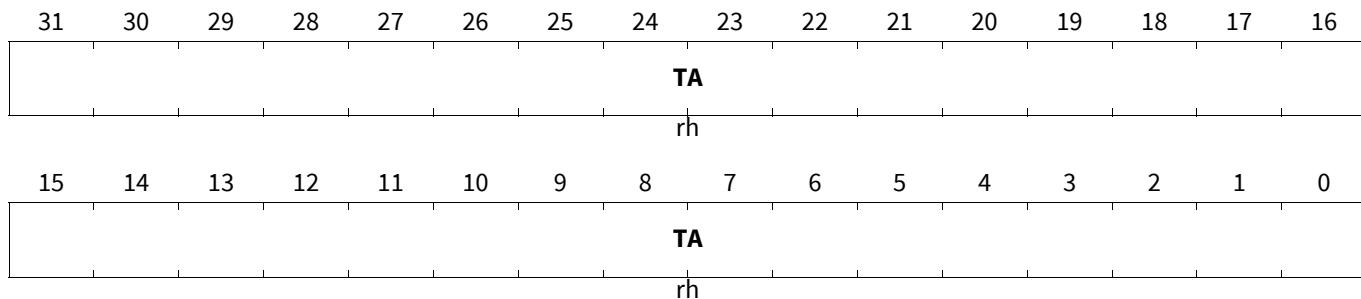
Application Reset Value:  $0000\ 0000_{\text{H}}$

CPU\_PIEAR

Short address for domain CSFR

( $09210_{\text{H}}$ )

Application Reset Value:  $0000\ 0000_{\text{H}}$



Field	Bits	Type	Description
<b>TA</b>	31:0	rh	<b>Transaction Address</b> Physical address being accessed by operation that encountered program integrity error.

## CPU Subsystem

### Data Integrity Error Information Registers

Two architecturally visible registers (DIETR, DIEAR) allow software to localise the source of the last detected uncorrectable data memory integrity error.

These registers are updated when an uncorrectable data integrity error condition is detected and the DIETR.IED bit is zero. On update the DIETR.IED bit is set to one and remains set until cleared by software. Whilst DIETR.IED is set further hardware updates of DIETR and DIEAR are inhibited. The various error scenarios are defined below.

#### Data Integrity errors during load from data memory

When an error is detected during a load from a data memory the IE\_S, IE\_C, IE\_T and IE\_LPB, IE\_DLMU bits are updated to denote in which memory structure the error was detected. If the IE\_C bit is set the E\_INFO field will be updated with the cache way. The DIEAR register is updated with the address of the access. If the error detected is an uncorrectable error the IE\_UNC bit is set. The IED bit is set and all other DIETR register bits are set to zero. Note that a memory read may also be generated by a sub word write operation or a cache line eviction.

#### Data Integrity errors during a memory read initiated by an external access

When an error is detected during an external bus read from data memory the IE\_S, IE\_C, IE\_DLMU bits are updated to denote in which memory structure the error was detected. The IE\_BS is set and E\_INFO updated. The DIEAR register is updated with the address of the access. If the error detected is an uncorrectable error the IE\_UNC bit is set. The IED bit is set and all other DIETR register bits are set to zero. Note that a memory read may also be generated by a sub word write operation or a cache line eviction. The IE\_C bit can only ever be set if the cache is mapped into memory in SIST mode. In this case the E\_INFO field indicates the tag of the requesting master.

#### Data Integrity errors due to safety protection

When a safety protection violation is detected during a data memory access the IE\_SP, IE\_BS bits are set and the E\_INFO field updated. The DIEAR register is updated with the address of the access. The IED bit is set and all other DIETR register bits are set to zero.

#### Data Integrity errors due ECC errors at the bus interface

When an ECC error is detected at the data bus interface (either master or slave) the IE\_BI bit is set. If the error is during an external access the IE\_BS bit is set and the E\_INFO fields updated. The DIEAR register is updated with the address of the access. If the error detected is an uncorrectable error the IE\_UNC bit is set. The IED bit is set and all other DIETR register bits are set to zero.

#### Data integrity error - Memory Test Mode Violation

The DIETR.IE\_MTMV bit is set if the cache controller attempt to access the DMEM (Dcache/DSPR), Dtag or DLMU memories while they are in a test mode.

## CPU Subsystem

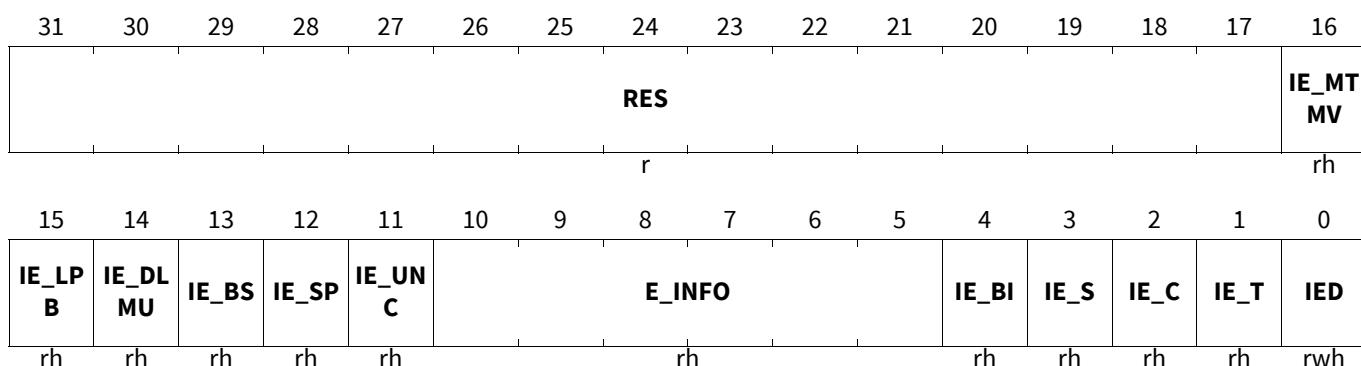
### CPUx Data Integrity Error Trap Register

#### DIETR

**CPUx Data Integrity Error Trap Register** **(19024<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**

#### CPU\_DIETR

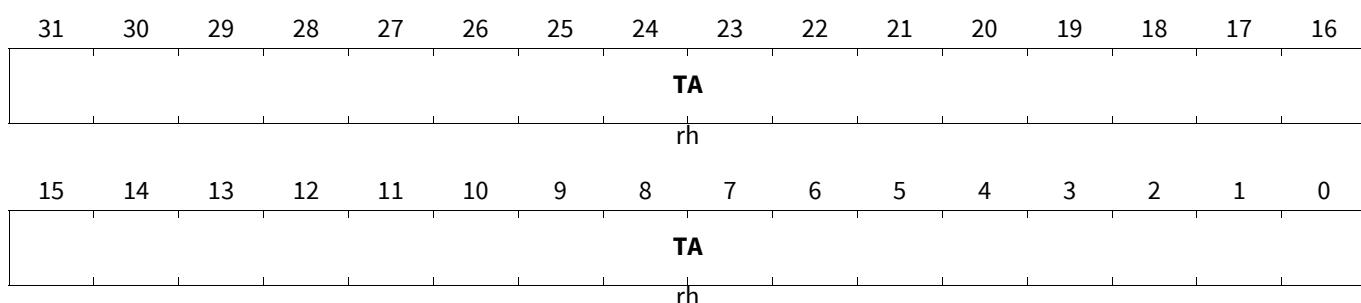
**Short address for domain CSFR** **(09024<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>IED</b>	0	rwh	<b>Integrity Error Detected</b> 0 <sub>B</sub> Write: Clear IED bit, re-enable DIETR and DIEAR update. Read : No data integrity error condition occurred 1 <sub>B</sub> Write : No Effect. Read: Data integrity error condition detected. DIETR and DIEAR contents valid, further DIETR and DIEAR updates disabled..
<b>IE_T</b>	1	rh	<b>Integrity Error - Tag Memory</b>
<b>IE_C</b>	2	rh	<b>Integrity Error - Cache Memory</b>
<b>IE_S</b>	3	rh	<b>Integrity Error - Scratchpad Memory</b>
<b>IE_BI</b>	4	rh	<b>Integrity Error - Bus Interface</b>
<b>E_INFO</b>	10:5	rh	<b>Error Information</b> If IE_BS = 1: Bus Master Tag ID of requesting masterIf IE_C = 1: Cache way.
<b>IE_UNC</b>	11	rh	<b>Dual Bit Error Detected</b>
<b>IE_SP</b>	12	rh	<b>Safety Protection Error Detected</b>
<b>IE_BS</b>	13	rh	<b>Bus Slave Access Indicator</b>
<b>IE_DLMU</b>	14	rh	<b>Integrity Error - DLMU</b>
<b>IE_LP_B</b>	15	rh	<b>Integrity Error - Local Pflash Bank</b>
<b>IE_MT_MV</b>	16	rh	<b>Memory Test Mode Violation detected</b>
<b>RES</b>	31:17	r	<b>Reserved</b> Read as 0; should be written with 0.

### CPUx Data Integrity Error Address Register

This register contains the physical address accessed by the operation that encountered a uncorrectable data memory integrity error. This register is only updated if DIETR.IED is zero.

**CPU Subsystem****DIEAR****CPUx Data Integrity Error Address Register (19020<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>****CPU\_DIEAR****Short address for domain CSFR (09020<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>TA</b>	31:0	rh	<b>Transaction Address</b> Physical address being accessed by operation that encountered data integrity error.

## CPU Subsystem

### SIST (Software In-System) Test Support

The CPU protects against memory integrity errors by ECC protection of the local CPU memories. This has the side-effect of requiring memory blocks wider than the normal data access path to the memory. The additional ECC storage bits are not easily accessible via the existing data paths, causing problems where software based testing of the memories is required. The CPU memories also include embedded memory arrays, such as the tag memories, which are not ordinarily accessible by the usual CPU datapaths. In order to address this problem, the CPUs include modes allowing all local memory arrays to be accessed, both as a backup for SSH based memory test and to allow the test and debug of the fault tolerant memory systems.

Each memory may be accessed either in normal operation, data array only or ECC check array only. This is controlled by the ECCS registers of the associated SSH controller.

The IODT bit controls read/write operation ordering. In normal operation (IODT=0) non-dependent read operations may overtake write operations. When SMACON.IODT is set all memory operations are performed in program order.

The SMACON register is protected by the safety\_endinit signal.

The mapping of cache and tag memories to the TriCore address space is controlled by the MTU\_MEMMAP register. See the MTU chapter for details.

### CPUx SIST Mode Access Control Register

#### SMACON

**CPUx SIST Mode Access Control Register**

**(1900C<sub>H</sub>)**

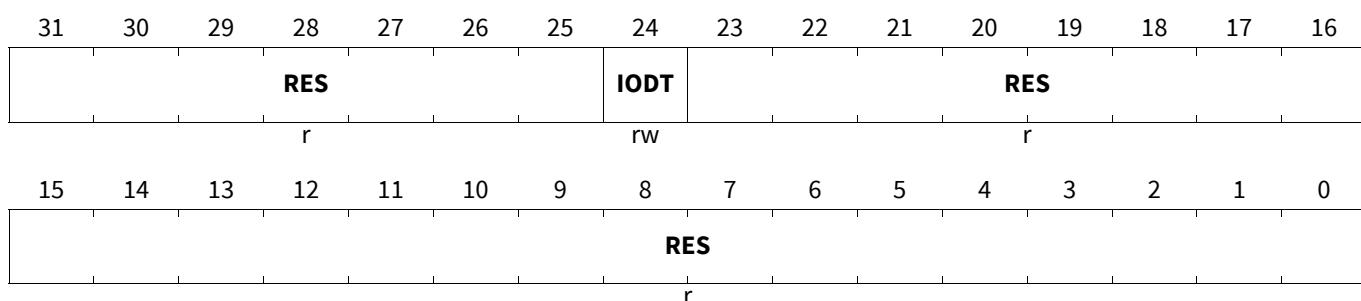
**Application Reset Value: 0000 0000<sub>H</sub>**

**CPU\_SMACON**

**Short address for domain CSFR**

**(0900C<sub>H</sub>)**

**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>RES</b>	23:0, 31:25	r	<b>Reserved</b> Read as 0; should be written with 0.
<b>IODT</b>	24	rw	<b>In-Order Data Transactions</b> 0 <sub>B</sub> Normal operation, Non-dependent loads bypass stores. 1 <sub>B</sub> In-order operation, Loads always flush preceding stores, processor store buffer disabled.

---

## CPU Subsystem

### 5.3.4.21 CPU Core Debug and Performance Counter Registers

The CPU Core Debug and performance counter registers are available for debug purposes. For a complete description of all registers, refer to the TriCore Architecture Manual.

#### 5.3.4.21.1 Counter Source Details

Details of the Performance counter sources is given below.

##### **IP\_DISPATCH\_STALL**

The counter is incremented on every cycle in which the Integer dispatch unit is stalled for whatever reason.

##### **LS\_DISPATCH\_STALL**

The counter is incremented on every cycle in which the Load-Store dispatch unit is stalled for whatever reason.

##### **LP\_DISPATCH\_STALL**

The counter is incremented on every cycle in which the Loop dispatch unit is stalled for whatever reason.

##### **PCACHE\_HIT**

The counter is incremented whenever the target of a cached fetch request from the fetch unit is found in the program cache.

##### **PCACHE\_MISS**

The counter is incremented whenever the target of a cached fetch request from the fetch unit is not found in the program cache and hence a bus fetch is initiated.

##### **MULTI\_ISSUE**

The counter is incremented in any cycle where more than one instruction is issued.

##### **DCACHE\_HIT**

The counter is incremented whenever the target of a cached request from the Load-Store unit is found in the data cache.

##### **DRB\_HIT**

The counter is incremented whenever the target of a cached request from the Load-Store unit is found in the data read buffer.

##### **DCACHE\_MISS\_CLEAN**

The counter is incremented whenever the target of a cached request from the Load-Store unit is not found in the data cache and hence a bus fetch is initiated with no dirty cache line eviction.

##### **DRB\_MISS**

The counter is incremented whenever the target of a cached request from the Load-Store unit is not found in the data read buffer and hence a bus fetch is initiated.

##### **DCACHE\_MISS\_DIRTY**

The counter is incremented whenever the target of a cached request from the Load-Store unit is not found in the data cache and hence a bus fetch is initiated with the writeback of a dirty cache line.

## CPU Subsystem

### TOTAL\_BRANCH

The counter is incremented in any cycle in which a branch instruction is in a branch resolution stage of the pipeline (IP\_EX1, LS\_DEC, LP\_DEC).

### PMEM\_STALL

The counter is incremented whenever the fetch unit is requesting an instruction and the Instruction memory is stalled for whatever reason.

### DMEM\_STALL

The counter is incremented whenever the Load-Store unit is requesting a data operation and the data memory is stalled for whatever reason.

## Performance Counter Registers

The various counter sources listed above may be selected for counting by programming the M1, M2 and M3 fields in the CCTRL register. (See the TriCore architecture manual for register details). The mappings are listed below.

Register	Description	Offset Address
CCTRL	Counter Control Register.	FC00 <sub>H</sub>
CCNT	CPU Clock Count Register.	FC04 <sub>H</sub>
ICNT	Instruction Count Register.	FC08 <sub>H</sub>
M1CNT	Multi Count Register 1.	FC0C <sub>H</sub>
M2CNT	Multi Count Register 2.	FC10 <sub>H</sub>
M3CNT	Multi Count Register 3.	FC14 <sub>H</sub>

**Table 100 MultiCount Configuration**

CCTRL M1/M2/M3	M1CNT Count Function	M2CNT Count Function	M3CNT Count Function
000	IP_DISPATCH_STALL	LS_DISPATCH_STALL	LP_DISPATCH_STALL
001	PCACHE_HIT	PCACHE_MISS	MULTI_ISSUE
010	DCACHE_HIT	DCACHE_MISS_CLEAN	DCACHE_MISS_DIRTY
011	TOTAL_BRANCH	PMEM_STALL	DMEM_STALL

---

**CPU Subsystem****5.3.4.22 CPU Subsystem Register Summary**

This section summaries the CPU Subsystem registers. For complete descriptions of all registers refer to the TriCore Architecture Manual.

To increase performance all stores to CSFR or SFR register locations from the SRI are posted writes and complete silently. SRI stores to non-existent CSFR or SFR locations are not errored.

A disallowed access either via the SRI or with MTCR/MFCR to any CPU register (e.g. attempted write to read only register, attempted user mode access to SV, attempted access to E without Endinit, etc.) will NOT result in an Error.

**CPU Subsystem****5.3.4.22.1 Summary of CSFR Reset Values and Access Modes**

This section summarizes the reset values and access modes of the CPU CSFR registers.

**Table 101 Register Overview - CORE\_SPECIAL\_FUNCTION\_REGISTERS (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
TASK_ASI	CPUx Task Address Space Identifier Register	18004 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">25</a>
PMA0	CPUx Data Access CacheabilityRegister	18100 <sub>H</sub>	U,SV,32	SV,32,P,CEx	Application Reset	<a href="#">27</a>
PMA1	CPUx Code Access CacheabilityRegister	18104 <sub>H</sub>	U,SV,32	SV,32,P,CEx	Application Reset	<a href="#">28</a>
PMA2	CPUx Peripheral Space Identifier register	18108 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">29</a>
COMPAT	CPUx Compatibility Control Register	19400 <sub>H</sub>	U,SV,32	SV,32,P,SE	Application Reset	<a href="#">30</a>
PCXI	CPUx Previous Context Information Register	1FE00 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">24</a>
PSW	CPUx Program Status Word	1FE04 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">22</a>
PC	CPUx Program Counter	1FE08 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">116</a>
SYSCON	CPUx System Configuration Register	1FE14 <sub>H</sub>	U,SV,32	SV,32,P,SE	See page <a href="#">115</a>	<a href="#">115</a>
CPU_ID	CPUx Identification Register TC1.6.2P	1FE18 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">26</a>
CORE_ID	CPUx Core Identification Register	1FE1C <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">26</a>
BIV	CPUx Base Interrupt Vector Table Pointer	1FE20 <sub>H</sub>	U,SV,32	SV,32,P,CEx	Application Reset	<a href="#">116</a>
BTV	CPUx Base Trap Vector Table Pointer	1FE24 <sub>H</sub>	U,SV,32	SV,32,P,CEx	Application Reset	<a href="#">117</a>
ISP	CPUx Interrupt Stack Pointer	1FE28 <sub>H</sub>	U,SV,32	SV,32,P,CEx	Application Reset	<a href="#">117</a>
ICR	CPUx Interrupt Control Register	1FE2C <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">115</a>
FCX	CPUx Free CSA List Head Pointer	1FE38 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">118</a>
LCX	CPUx Free CSA List Limit Pointer	1FE3C <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">118</a>
CUS_ID	CPUx Customer ID register	1FE50 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">27</a>

**CPU Subsystem****Table 102 Register Overview - GPR\_REGISTERS (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
Dy	CPUx Data General Purpose Register y	1FF00 <sub>H</sub> +y*4	U,SV,32	SV,32,P	Application Reset	<a href="#">31</a>
Ay	CPUx Address General Purpose Register y	1FF80 <sub>H</sub> +y*4	U,SV,32	SV,32,P	Application Reset	<a href="#">31</a>

**Table 103 Register Overview - MEMORY\_PROTECTION\_REGISTERS (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
DPRy_L	CPUx Data Protection Range y, Lower Bound Register	1C000 <sub>H</sub> +y*8	U,SV,32	SV,32,P	Application Reset	<a href="#">119</a>
DPRy_U	CPUx Data Protection Range y, Upper Bound Register	1C004 <sub>H</sub> +y*8	U,SV,32	SV,32,P	Application Reset	<a href="#">119</a>
CPRy_L	CPUx Code Protection Range y Lower Bound Register	1D000 <sub>H</sub> +y*8	U,SV,32	SV,32,P	Application Reset	<a href="#">120</a>
CPRy_U	CPUx Code Protection Range y Upper Bound Register	1D004 <sub>H</sub> +y*8	U,SV,32	SV,32,P	Application Reset	<a href="#">120</a>
CPXE_y	CPUx Code Protection Execute Enable Register Set y	1E000 <sub>H</sub> +y*4	U,SV,32	SV,32,P	Application Reset	<a href="#">121</a>
DPRE_y	CPUx Data Protection Read Enable Register Set y	1E010 <sub>H</sub> +y*4	U,SV,32	SV,32,P	Application Reset	<a href="#">121</a>
DPWE_y	CPUx Data Protection Write Enable Register Set y	1E020 <sub>H</sub> +y*4	U,SV,32	SV,32,P	Application Reset	<a href="#">122</a>
CPXE_y	CPUx Code Protection Execute Enable Register Set y	1E040 <sub>H</sub> +(y-4)*4	U,SV,32	SV,32,P	Application Reset	<a href="#">121</a>
DPRE_y	CPUx Data Protection Read Enable Register Set y	1E050 <sub>H</sub> +(y-4)*4	U,SV,32	SV,32,P	Application Reset	<a href="#">121</a>
DPWE_y	CPUx Data Protection Write Enable Register Set y	1E060 <sub>H</sub> +(y-4)*4	U,SV,32	SV,32,P	Application Reset	<a href="#">122</a>

## CPU Subsystem

**Table 104 Register Overview - TEMPORAL\_PROTECTION\_REGISTERS (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
TPS_CON	CPUx Temporal Protection System Control Register	1E400 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	38
TPS_TIMERy	CPUx Temporal Protection System Timer Register y	1E404 <sub>H</sub> + y*4	U,SV,32	SV,32,P	Application Reset	39
TPS_EXTIM_ENT_RY_LVAL	CPUx Exception Entry Timer Load Value	1E440 <sub>H</sub>	U,SV,32	SV,32,SE,P	Application Reset	42
TPS_EXTIM_ENT_RY_CVAL	CPUx Exception Entry Timer Current Value	1E444 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	43
TPS_EXTIM_EXIT_LVAL	CPUx Exception Exit Timer Load Value	1E448 <sub>H</sub>	U,SV,32	SV,32,SE,P	Application Reset	42
TPS_EXTIM_EXIT_CVAL	CPUx Exception Exit Timer Current Value	1E44C <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	43
TPS_EXTIM_CLASS_EN	CPUx Exception Timer Class Enable Register	1E450 <sub>H</sub>	U,SV,32	SV,32,SE,P	Application Reset	44
TPS_EXTIM_STATUS	CPUx Exception Timer Status Register	1E454 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	44
TPS_EXTIM_FCX	CPUx Exception Timer FCX Register	1E458 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	45

**Table 105 Register Overview - FLOATING\_POINT\_SPECIAL\_FUNCTION\_REGISTERS (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
FPU_TRAP_CON	CPUx Trap Control Register	1A000 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	32
FPU_TRAP_PC	CPUx Trapping Instruction Program Counter Register	1A004 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	33
FPU_TRAP_OPC	CPUx Trapping Instruction Opcode Register	1A008 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	34
FPU_TRAP_SRC1	CPUx Trapping Instruction Operand Register	1A010 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	35
FPU_TRAP_SRC2	CPUx Trapping Instruction Operand Register	1A014 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	35
FPU_TRAP_SRC3	CPUx Trapping Instruction Operand Register	1A018 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	36

## CPU Subsystem

**Table 106 Register Overview - CORE\_DEBUG\_PERFORMANCE\_COUNTER\_REGISTERS (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
TRiEVT	CPUx Trigger Event i	1F000 <sub>H</sub> +i *8	U,SV,32	SV,32,P	Debug Reset	<a href="#">115</a>
TRiADR	CPUx Trigger Address i	1F004 <sub>H</sub> +i *8	U,SV,32	SV,32,P	Debug Reset	<a href="#">115</a>
CCTRL	CPUx Counter Control	1FC00 <sub>H</sub>	U,SV,32	SV,32,P	Debug Reset	<a href="#">115</a>
CCNT	CPUx CPU Clock Cycle Count	1FC04 <sub>H</sub>	U,SV,32	SV,32,P	Debug Reset	<a href="#">115</a>
ICNT	CPUx Instruction Count	1FC08 <sub>H</sub>	U,SV,32	SV,32,P	Debug Reset	<a href="#">115</a>
M1CNT	CPUx Multi-Count Register 1	1FC0C <sub>H</sub>	U,SV,32	SV,32,P	Debug Reset	<a href="#">115</a>
M2CNT	CPUx Multi-Count Register 2	1FC10 <sub>H</sub>	U,SV,32	SV,32,P	Debug Reset	<a href="#">115</a>
M3CNT	CPUx Multi-Count Register 3	1FC14 <sub>H</sub>	U,SV,32	SV,32,P	Debug Reset	<a href="#">115</a>
DBGSR	CPUx Debug Status Register	1FD00 <sub>H</sub>	U,SV,32	SV,32,P	Debug Reset	<a href="#">115</a>
EXEVT	CPUx External Event Register	1FD08 <sub>H</sub>	U,SV,32	SV,32,P	Debug Reset	<a href="#">115</a>
CREVT	CPUx Core Register Access Event	1FD0C <sub>H</sub>	U,SV,32	SV,32,P	Debug Reset	<a href="#">115</a>
SWEVT	CPUx Software Debug Event	1FD10 <sub>H</sub>	U,SV,32	SV,32,P	Debug Reset	<a href="#">115</a>
TRIG_ACC	CPUx TriggerAddressx	1FD30 <sub>H</sub>	U,SV,32	SV,32,P	Debug Reset	<a href="#">115</a>
DMS	CPUx Debug Monitor Start Address	1FD40 <sub>H</sub>	U,SV,32	SV,32,P	Debug Reset	<a href="#">122</a>
DCX	CPUx Debug Context Save Area Pointer	1FD44 <sub>H</sub>	U,SV,32	SV,32,P	Debug Reset	<a href="#">123</a>
DBGTCR	CPUx Debug Trap Control Register	1FD48 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">115</a>

**Table 107 Register Overview - DMI\_REGISTERS (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
SEGEN	CPUx SRI Error Generation Register	11030 <sub>H</sub>	U,SV,32	SV,32,P,CEx	Application Reset	<a href="#">128</a>
DCON2	CPUx Data Control Register 2	19000 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">90</a>
DSTR	CPUx Data Synchronous Trap Register	19010 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">91</a>
DATR	CPUx Data Asynchronous Trap Register	19018 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">92</a>

**CPU Subsystem****Table 107 Register Overview - DMI\_REGISTERS (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
DEADD	CPUx Data Error Address Register	1901C <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	93
DCON0	CPUx Data Memory Control Register	19040 <sub>H</sub>	U,SV,32	SV,32,P,CEx	Application Reset	90

**Table 108 Register Overview - PMI\_REGISTERS (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
PSTR	CPUx Program Synchronous Trap Register	19200 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	84
PCON1	CPUx Program Control 1	19204 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	83
PCON2	CPUx Program Control 2	19208 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	84
PCON0	CPUx Program Control 0	1920C <sub>H</sub>	U,SV,32	SV,32,P,CEx	Application Reset	83

**CPU Subsystem****5.3.4.22.2 Summary of SFR Reset Values and Access modes**

This section summarizes the reset values and access modes of the CPU SFR registers.

**Table 109 Register Overview - SPR (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
SPR_SPROT_RG_NLAi	CPUx Safety Protection SPR Region Lower Address Register i	0E000 <sub>H</sub> +i *10 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<a href="#">129</a>
SPR_SPROT_RG_NUAI	CPUx Safety Protection SPR Region Upper Address Register i	0E004 <sub>H</sub> +i *10 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<a href="#">129</a>
SPR_SPROT_RG_NACCENAi_W	CPUx Safety Protection SPR Region Write Access Enable Register Ai	0E008 <sub>H</sub> +i *10 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<a href="#">130</a>
SPR_SPROT_RG_NACCENBi_W	CPUx Safety Protection SPR Region Write Access Enable Register Bi	0E00C <sub>H</sub> +i *10 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<a href="#">130</a>
SPR_SPROT_RG_NACCENAi_R	CPUx Safety Protection SPR Region Read Access Enable Register Ai	0E088 <sub>H</sub> +i *10 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<a href="#">131</a>
SPR_SPROT_RG_NACCENBi_R	CPUx Safety Protection SPR Region Read Access Enable Register Bi	0E08C <sub>H</sub> +i *10 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<a href="#">131</a>

**Table 110 Register Overview - DLMU (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
DLMU_SPROT_R_GNLAI	CPUx Safety Protection DLMU Region Lower Address Register i	0E200 <sub>H</sub> +i *10 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<a href="#">132</a>
DLMU_SPROT_R_GNUAI	CPUx Safety protection DLMU Region Upper Address Register i	0E204 <sub>H</sub> +i *10 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<a href="#">133</a>
DLMU_SPROT_R_GNACCENAi_W	CPUx Safety Protection Region DLMU Write Access Enable Register Ai	0E208 <sub>H</sub> +i *10 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<a href="#">133</a>
DLMU_SPROT_R_GNACCENBi_W	CPUx Safety Protection Region DLMU Write Access Enable Register Bi	0E20C <sub>H</sub> +i *10 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<a href="#">134</a>

**CPU Subsystem****Table 110 Register Overview - DLMU (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
DLMU_SPROT_R GNACCENAI_R	CPUx Safety Protection Region DLMU Read Access Enable Register Ai	0E288 <sub>H</sub> +i *10 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<b>134</b>
DLMU_SPROT_R GNACCENBi_R	CPUx Safety Protection Region DLMU Read Access Enable Register Bi	0E28C <sub>H</sub> +i *10 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<b>135</b>

**Table 111 Register Overview - SAFETY\_REGISTER\_PROTECTION (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
SFR_SPROT_ACC ENA_W	CPUx Safety Protection Register Access Enable Register A	0E100 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<b>137</b>
SFR_SPROT_ACC ENB_W	CPUx Safety Protection Region Access Enable Register B	0E104 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<b>137</b>
LPB_SPROT_ACC ENA_R	CPUx Safety Protection Region LPB Read Access Enable Register A	0E110 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<b>135</b>
LPB_SPROT_ACC ENB_R	CPUx Safety Protection Region LPB Read Access Enable Register B	0E114 <sub>H</sub>	U,SV,32	SV,32,SE	Application Reset	<b>136</b>

**Table 112 Register Overview - KERNEL\_RESET\_REGISTERS (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
KRST0	CPUx Reset Register 0	0D000 <sub>H</sub>	U,SV,32	SV,32,E,P	Application Reset	<b>19</b>
KRST1	CPUx Reset Register 1	0D004 <sub>H</sub>	U,SV,32	SV,32,E,P	Application Reset	<b>20</b>
KRSTCLR	CPUx Reset Clear Register	0D008 <sub>H</sub>	U,SV,32	SV,32,E,P	Application Reset	<b>19</b>

## CPU Subsystem

**Table 113 Register Overview - FLASH\_CONFIGURATION\_REGISTERS (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
FLASHCON0	CPUx Flash Configuration Register 0	01100 <sub>H</sub>	U,SV,32	SV,32,P,E	See page 95	95
FLASHCON1	CPUx Flash Configuration Register 1	01104 <sub>H</sub>	U,SV,32	SV,32,P,E	Application Reset	95
FLASHCON2	CPUx Flash Configuration Register 2	01108 <sub>H</sub>	U,SV,32	SV,32,P,E	Application Reset	96
FLASHCON3	CPUx Flash Configuration Register 3	0110C <sub>H</sub>	U,SV,32	SV,32,P,E	Application Reset	98
FLASHCON4	CPUx Flash Configuration Register 4	01110 <sub>H</sub>	U,SV,32	SV,32,P,ST	Application Reset	100

---

## CPU Subsystem

### 5.3.5 CPU Instruction Timing

This section gives information on CPU instruction timing by execution units.

#### Definition of Terms:

- **Repeat Rate**

Assuming the same instruction is being issued sequentially, repeat is the minimum number of clock cycles between two consecutive issues. There may be additional delays described elsewhere due to internal pipeline effects when issuing a different subsequent instruction.

- **Result Latency**

The number of clock cycles from the cycle when the instruction is issued to the cycle when the result value is available to be used as an operand to a subsequent instruction or written into a GPR. Result latency is not meaningful for instructions that do not write a value into a GPR.

- **Address Latency**

The number of clock cycles from the cycle when the instruction is issued to the cycle when the addressing mode updated value is available as an operand to a subsequent instruction or written into an Address Register.

- **Flow Latency**

The number of clock cycles from the cycle when the instruction is issued to the cycle when the next instruction (located at the target location or the next sequential instruction if the control change is conditional) is issued.

## CPU Subsystem

## 5.3.5.1 Integer-Pipeline Instructions

These are the Integer-Pipeline instruction timings for each instruction.

## 5.3.5.1.1 Simple Arithmetic Instruction Timings

Each instruction is single issued.

**Table 114 Simple Arithmetic Instruction Timing**

Instruction	Result Latency	Repeat Rate	Instruction	Result Latency	Repeat Rate
<b>Integer Pipeline Arithmetic Instructions</b>					
<b>ABS</b>	1	1	<b>MAX.H</b>	1	1
<b>ABS.B</b>	1	1	<b>MAX.HU</b>	1	1
<b>ABS.H</b>	1	1	<b>MAX.U</b>	1	1
<b>ABSDIF</b>	1	1	<b>MIN</b>	1	1
<b>ABSDIF.B</b>	1	1	<b>MIN.B</b>	1	1
<b>ABSDIF.H</b>	1	1	<b>MIN.BU</b>	1	1
<b>ABSDIFS</b>	2	1	<b>MIN.H</b>	1	1
<b>ABSDIFS.H</b>	2	1	<b>MIN.HU</b>	1	1
<b>ABSS</b>	2	1	<b>MIN.U</b>	1	1
<b>ABSS.H</b>	2	1	<b>RSUB</b>	1	1
<b>ADD</b>	1	1	<b>RSUBS</b>	2	1
<b>ADD.B</b>	1	1	<b>RSUBS.U</b>	2	1
<b>ADD.H</b>	1	1	<b>SAT.B</b>	1	1
<b>ADDC</b>	1	1	<b>SAT.BU</b>	1	1
<b>ADDI</b>	1	1	<b>SAT.H</b>	1	1
<b>ADDIH</b>	1	1	<b>SAT.HU</b>	1	1
<b>ADDS</b>	2	1	<b>SEL</b>	1	1
<b>ADDS.H</b>	2	1	<b>SELN</b>	1	1
<b>ADDS.HU</b>	2	1	<b>SUB</b>	1	1
<b>ADDS.U</b>	2	1	<b>SUB.B</b>	1	1
<b>ADDX</b>	1	1	<b>SUB.H</b>	1	1
<b>CADD</b>	1	1	<b>SUBC</b>	1	1
<b>CADDN</b>	1	1	<b>SUBS</b>	2	1
<b>CSUB</b>	1	1	<b>SUBS.H</b>	2	1
<b>CSUBN</b>	1	1	<b>SUBS.HU</b>	2	1
<b>MAX</b>	1	1	<b>SUBS.U</b>	2	1
<b>MAX.B</b>	1	1	<b>SUBX</b>	1	1
<b>MAX.BU</b>	1	1	<b>SHUFFLE</b>	2	1
<b>POPCNT</b>	2	1			

## Compare Instructions

## CPU Subsystem

Table 114 Simple Arithmetic Instruction Timing (cont'd)

Instruction	Result Latency	Repeat Rate	Instruction	Result Latency	Repeat Rate
<b>EQ</b>	1	1	<b>LT.B</b>	1	1
<b>EQ.B</b>	1	1	<b>LT.BU</b>	1	1
<b>EQ.H</b>	1	1	<b>LT.H</b>	1	1
<b>EQ.W</b>	1	1	<b>LT.HU</b>	1	1
<b>EQANY.B</b>	1	1	<b>LT.U</b>	1	1
<b>EQANY.H</b>	1	1	<b>LT.W</b>	1	1
<b>GE</b>	1	1	<b>LT.WU</b>	1	1
<b>GE.U</b>	1	1	<b>NE</b>	1	1
<b>LT</b>	1	1			
<b>Count Instructions</b>					
<b>CLO</b>	1	1	<b>CLS.H</b>	1	1
<b>CLO.H</b>	1	1	<b>CLZ</b>	1	1
<b>CLS</b>	1	1	<b>CLZ.H</b>	1	1
<b>Extract Instructions</b>					
<b>DEXTR</b>	2	1	<b>INS.T</b>	1	1
<b>EXTR</b>	2	1	<b>INSN.T</b>	1	1
<b>EXTR.U</b>	2	1	<b>INSERT</b>	2	1
<b>IMASK</b>	2	1			
<b>Logical Instructions</b>					
<b>AND</b>	1	1	<b>OR.EQ</b>	1	1
<b>AND.AND.T</b>	1	1	<b>OR.GE</b>	1	1
<b>AND.ANDN.T</b>	1	1	<b>OR.GE.U</b>	1	1
<b>AND.EQ</b>	1	1	<b>OR.LT</b>	1	1
<b>AND.GE</b>	1	1	<b>OR.LT.U</b>	1	1
<b>AND.GE.U</b>	1	1	<b>OR.NE</b>	1	1
<b>AND.LT</b>	1	1	<b>OR.NOR.T</b>	1	1
<b>AND.LT.U</b>	1	1	<b>OR.OR.T</b>	1	1
<b>AND.NE</b>	1	1	<b>OR.T</b>	1	1
<b>AND.NOR.T</b>	1	1	<b>ORN</b>	1	1
<b>AND.OR.T</b>	1	1	<b>ORN.T</b>	1	1
<b>AND.T</b>	1	1	<b>XNOR</b>	1	1
<b>ANDN</b>	1	1	<b>XNOR.T</b>	1	1
<b>ANDN.T</b>	1	1	<b>XOR</b>	1	1
<b>NAND</b>	1	1	<b>XOR.EQ</b>	1	1
<b>NAND.T</b>	1	1	<b>XOR.GE</b>	1	1
<b>NOR</b>	1	1	<b>XOR.GE.U</b>	1	1
<b>NOR.T</b>	1	1	<b>XOR.LT</b>	1	1

## CPU Subsystem

Table 114 Simple Arithmetic Instruction Timing (cont'd)

Instruction	Result Latency	Repeat Rate	Instruction	Result Latency	Repeat Rate
<b>OR</b>	1	1	<b>XOR.LT.U</b>	1	1
<b>OR.AND.T</b>	1	1	<b>XOR.NE</b>	1	1
<b>OR.ANDN.T</b>	1	1	<b>XOR.T</b>	1	1
<b>Move Instructions</b>					
<b>CMOV</b>	1	1	<b>MOV.U</b>	1	1
<b>CMOVN</b>	1	1	<b>MOVH</b>	1	1
<b>MOV (32 Bit)</b>	1	1	<b>MOV (64bit)</b>	2	1
<b>Shift Instructions</b>					
<b>SH</b>	1	1	<b>SH.NE</b>	1	1
<b>SH.AND.T</b>	1	1	<b>SH.NOR.T</b>	1	1
<b>SH.ANDN.T</b>	1	1	<b>SH.OR.T</b>	1	1
<b>SH.EQ</b>	1	1	<b>SH.ORN.T</b>	1	1
<b>SH.GE</b>	1	1	<b>SH.XNOR.T</b>	1	1
<b>SH.GE.U</b>	1	1	<b>SH.XOR.T</b>	1	1
<b>SH.H</b>	1	1	<b>SHA</b>	1	1
<b>SH.LT</b>	1	1	<b>SHA.H</b>	1	1
<b>SH.LT.U</b>	1	1	<b>SHAS</b>	2	1
<b>SH.NAND.T</b>	1	1			
<b>Coprocessor 0 Instructions</b>					
<b>BMERGE</b>	2	1	<b>IXMIN</b>	2	1
<b>BSPLIT</b>	2	1	<b>UNPACK</b>	2	1
<b>PARITY</b>	2	1	<b>IXMAX</b>	2	1
<b>PACK</b>	2	1	<b>IXMAX.U</b>	2	1
<b>IXMIN.U</b>	2	1	<b>CRC32B.W</b>	2	1
<b>CRC32L.W</b>	2	1	<b>CRC32.B</b>	2	1
<b>CRCN</b>	2	1			
<b>Integer Divide Instructions</b>					
<b>DVADJ</b>	2	1	<b>DVSTEP</b>	6	4
<b>DVINIT</b>	2	1	<b>DVSTEP.U</b>	6	4
<b>DVINIT.U</b>	2	1	<b>DIV</b>	4-11	3-9
<b>DVINIT.B</b>	2	1	<b>DIV.U</b>	4-11	3-9
<b>DVINIT.H</b>	2	1			
<b>DVINIT.BU</b>	2	1			
<b>DVINIT.HU</b>	2	1			

The latency and repeat rate values listed for the DIV and DIV.U instructions are the minimum and maximum values. The algorithm used allows for early termination of the instruction once the full result is available.

**CPU Subsystem****5.3.5.1.2 Multiply Instruction Timings**

Each instruction is single issued.

**Table 115 Multiply Instruction Timing**

Instruction	Result Latency	Repeat Rate	Instruction	Result Latency	Repeat Rate
<b>MUL</b>	2	1	<b>MUL.Q</b>	3	1
<b>MUL.U</b>	2	1	<b>MULM.H</b>	3	1
<b>MULS</b>	3	1	<b>MULR.H</b>	3	1
<b>MULS.U</b>	3	1	<b>MULR.Q</b>	3	1
<b>MUL.H</b>	3	1			

**5.3.5.1.3 Multiply Accumulate (MAC) Instruction Timing**

Each instruction is single issued.

**Table 116 Multiply Accumulate Instruction Timing**

Instruction	Result Latency	Repeat Rate	Instruction	Result Latency	Repeat Rate
<b>MADD</b>	3	1	<b>MSUB</b>	3	1
<b>MADD.U</b>	3	1	<b>MSUB.U</b>	3	1
<b>MADDS</b>	3	1	<b>MSUBS</b>	3	1
<b>MADDS.U</b>	3	1	<b>MSUBS.U</b>	3	1
<b>MADD.H</b>	3	1	<b>MSUB.H</b>	3	1
<b>MADD.Q</b>	3	1	<b>MSUB.Q</b>	3	1
<b>MADDM.H</b>	3	1	<b>MSUBM.H</b>	3	1
<b>MADDMS.H</b>	3	1	<b>MSUBMS.H</b>	3	1
<b>MADDR.H</b>	3	1	<b>MSUBR.H</b>	3	1
<b>MADDR.Q</b>	3	1	<b>MSUBR.Q</b>	3	1
<b>MADDRS.H</b>	3	1	<b>MSUBRS.H</b>	3	1
<b>MADDRS.Q</b>	3	1	<b>MSUBRS.Q</b>	3	1
<b>MADDS.H</b>	3	1	<b>MSUBS.H</b>	3	1
<b>MADDS.Q</b>	3	1	<b>MSUBS.Q</b>	3	1
<b>MADDSU.H</b>	3	1	<b>MSUBAD.H</b>	3	1
<b>MADDSUM.H</b>	3	1	<b>MSUBADM.H</b>	3	1
<b>MADDSUMS.H</b>	3	1	<b>MSUBADMS.H</b>	3	1
<b>MADDSUR.H</b>	3	1	<b>MSUBADR.H</b>	3	1
<b>MADDSURS.H</b>	3	1	<b>MSUBADRS.H</b>	3	1
<b>MADDSUS.H</b>	3	1	<b>MSUBADS.H</b>	3	1

For All MADD, MSUB and MUL type instructions the result latency is reduced to 1 for accumulator forwarding between similar instructions.

## CPU Subsystem

For MADD.Q, MADDS.Q, MSUB.Q, MSUBS.Q Instructions:

MADD.Q, MADDS.Q, MSUB.Q, MSUBS.Q	Result Latency	Repeat Rate
16 × 16	3	1
16 × 32	3	1
32 × 32	3	1

### 5.3.5.1.4 Control Flow Instruction Timing

Control flow instruction timing for TC1.6.2P is complicated by the use of branch target buffers and fetch FIFOs.

- Incorrectly predicted LS instructions incur a three cycle branch recovery penalty.
- Incorrectly predicted IP instructions incur a four cycle branch recovery penalty.
- Correctly predicted not taken branches incur no penalty
- Correctly predicted taken branch incur a penalty of up to two cycles depending on the state of the fetch FIFOs and the branch target buffer.
- Loop instructions incur the same penalty as an LS conditional jump instruction.

Assumptions

- All target locations yield a full instruction in one access (i.e. not 16-bits of a 32-bit instruction).
- All code fetches take a single cycle.
- Timing is best case; no cache misses for context operations, no pending stores.

**Table 117 Control flow timing**

Prediction-Result	Flow Latency (LS, LP)	Repeat rate (LS,LP)	Flow Latency (IP)	Repeat Rate (IP)
Correct Not-Taken	1	1	1	1
Correct Taken	1-2	1-2	1-2	1-2
Incorrect Not-Taken	3	4	3	4
Incorrect Taken	3	4	3	4

## CPU Subsystem

## 5.3.5.2 Load-Store Pipeline Instructions

This section summarizes the Load-Store Pipeline instructions.

## 5.3.5.2.1 Address Arithmetic Timing

Each instruction is single issued.

**Table 118 Address Arithmetic Instruction Timing**

Instruction	Result Latency	Repeat Rate	Instruction	Result Latency	Repeat Rate
<b>Load Store Arithmetic Instructions</b>					
<b>ADD.A</b>	1	1	<b>GE.A</b>	1	1
<b>ADDIH.A</b>	1	1	<b>LT.A</b>	1	1
<b>ADDSC.A</b>	2	1	<b>NE.A</b>	1	1
<b>ADDSC.AT</b>	2	1	<b>NEZ.A</b>	1	1
<b>EQ.A</b>	1	1	<b>SUB.A</b>	1	1
<b>EQZ.A</b>	1	1	<b>NOP</b>	1	1
<b>Trap and Interrupt Instructions</b>					
<b>DEBUG</b>	-	1	<b>TRAPS<sup>1)</sup></b>	-	1
<b>DISABLE</b>	-	1	<b>TRAPV<sup>1)</sup></b>	-	1
<b>ENABLE</b>	-	1	<b>RSTV</b>	-	1
<b>RESTORE</b>	-	1	<b>WAIT<sup>2)</sup></b>	-	1
<b>Move Instructions</b>					
<b>MFCR</b>	2	1	<b>MOV.A</b>	1	1
<b>MTCR</b>	1	1	<b>MOV.AA</b>	1	1
<b>MOVH.A</b>	1	1	<b>MOV.D</b>	1	1
<b>Sync Instructions</b>					
<b>DSYNC</b>	-	1	<b>ISYNC<sup>3)</sup></b>	-	1

- 1) Execution cycles when no TRAP is taken. The execution timing in the case of raising these TRAPs is the same as other TRAPs such as SYSCALL.
- 2) The latency of the WAIT instruction is hidden by the context save of the interrupt. Effective latency is zero.
- 3) Repeat rate assumes that code refetch takes a single cycle.

## CPU Subsystem

**5.3.5.2.2 CSA Control Flow Instruction Timing**

This section summarizes the timing of CSA Control Flow instructions.

- All targets yield a full instruction in one access (not 16-bits of a 32-bit instruction).
- All code fetches take a single cycle. Timing is best case; no cache misses for context operations, no pending stores.

**Table 119 CSA Control Flow Instruction Timing**

Instruction	Result Latency	Repeat Rate	Instruction	Result Latency	Repeat Rate
<b>CALL</b>	4-8	4-8	<b>SYSCALL</b>	4-8	4-8
<b>CALLA</b>	4-8	4-8	<b>SVLCX</b>	4-8	4-8
<b>CALLI</b>	4-8	4-8	<b>RSLCX</b>	4-8	4-8
<b>RET</b>	4-8	4-8	<b>RFE</b>	4-8	4-8
<b>BISR</b>	4-8	4-8	<b>RFM</b>	4-8	4-8
<b>FCALL</b>	1	1	<b>FCALLA</b>	1	1
<b>FCALLI</b>	1	1	<b>FRET</b>	1	1

Access to DSPR require 4 cycles, accesses to cached external memory require 8 cycles.

**5.3.5.2.3 Load Instruction Timing**

Load instructions can produce two results if they use the pre-increment, post-increment, circular or bit-reverse addressing modes. Hence, in those cases there are two latencies that must be specified, the result latency for the value loaded from memory and the address latency for using the updated address register result.

- Each instruction is single issued.
- The memory references is naturally aligned.
- The memory accessed takes a single cycle to return a data item.
- Timing is best case; no cache misses, no pending stores.

**Table 120 Load Instruction Timing**

Instruction	Address Latency	Result Latency	Repeat Rate	Instruction	Address Latency	Result Latency	Repeat Rate
<b>Load Instructions</b>							
<b>LD.A</b>	1	3	1	<b>LD.Q</b>	1	2	1
<b>LD.B</b>	1	2	1	<b>LD.W</b>	1	2	1
<b>LD.BU</b>	1	2	1	<b>LDLCX</b>	5-9	5-9	5-9
<b>LD.D</b>	1	2	1	<b>LDUCX</b>	5-9	5-9	5-9
<b>LD.DA</b>	1	3	1	<b>SWAP.W</b>	2	3	6
<b>LD.H</b>	1	2	1	<b>LEA<sup>1)</sup></b>	-	1	1
<b>LD.HU</b>	1	2	1	<b>CMPSWAP.W</b>	2	3	6
<b>SWAPMSK.W</b>	2	3	6	<b>LHA</b>	-	1	1

1) The addressing mode returning an updated address is not relevant for this instruction.

**CPU Subsystem****5.3.5.2.4 Store Instruction Timing**

Cache and Store instructions similar to Load instructions will have a result for the pre-increment, post-increment, circular or bit-reverse addressing modes, but do not produce a ‘memory’ result.

- Each instruction is single issued.
- The memory references is naturally aligned.
- The memory accessed takes a single cycle to accept a data item.
- Timing is best case; no cache misses, no pending stores.

**Table 121 Cache and Store Instruction Timing**

<b>Instruction</b>	<b>Address Latency</b>	<b>Repeat Rate</b>	<b>Instruction</b>	<b>Address Latency</b>	<b>Repeat Rate</b>
<b>Cache Instructions</b>					
<b>CACHEA.I</b>	1	1	<b>CACHEA.W<sup>1)</sup></b>	1	1
<b>CACHEA.W<sup>1)</sup></b>	1	1	<b>CACHEI.W<sup>1)</sup></b>	1	1
<b>CACHEI.W<sup>1)</sup></b>	1	1	<b>CACHEI.I</b>	1	1
<b>Store Instructions</b>					
<b>ST.A</b>	1	1	<b>ST.T</b>	-	6
<b>ST.B</b>	1	1	<b>ST.W</b>	1	1
<b>ST.D</b>	1	1	<b>STLCX</b>	5-9	5-9
<b>ST.DA</b>	1	1	<b>STUCX</b>	5-9	5-9
<b>ST.H</b>	1	1	<b>LDMST</b>	1	6
<b>ST.Q</b>	1	1			

- 1) Repeat rate assumes that no memory writeback operation occurs. Otherwise the repeat rate will depend upon the time for the castout buffers to clear.

**CPU Subsystem****5.3.5.3 Floating Point Pipeline Timing**

Each instruction is single issued.

**Table 122 Floating Point Instruction Timing**

Instruction	Result Latency	Repeat Rate	Instruction	Result Latency	Repeat Rate
<b>Floating Point Instructions</b>					
<b>ADD.F</b>	2	1	<b>ITOF</b>	2	1
<b>CMP.F</b>	1	1	<b>MADD.F</b>	3	1
<b>DIV.F</b>	8	6	<b>MSUB.F</b>	3	1
<b>FTOI</b>	2	1	<b>MUL.F</b>	2	1
<b>FTOIZ</b>	2	1	<b>Q31TOF</b>	2	1
<b>FTOQ31</b>	2	1	<b>QSEED.F</b>	1	1
<b>FTOQ31Z</b>	2	1	<b>SUB.F</b>	2	1
<b>FTOU</b>	2	1	<b>UPDFL</b>	-	1
<b>FTOUZ</b>	2	1	<b>UTOF</b>	2	1
<b>FTOHP</b>	2	1	<b>HPTOF</b>	2	1

**5.3.6 Local Memory Details**

This chapter describes the features of the local memories of the TC1.6.2 processor.

The processor implementation includes the following local memories. The sizes of each of these memories is detailed in Section 1-2

- Data Scratch Pad Ram (DSPR) - Single cycle memory for local data operations, code access via bus
- Data Cache (DCache) - Single cycle memory for data operations
- Distributed LMU (DLMU) - Single cycle memory for local data operations, code access via bus. Contiguous with other LMU memory in the system.
  - The DLMU may be implemented as a standard SRAM or as a Standby SRAM. Where standby SRAM is indicated as implemented the entire local DLMU contents will be maintained during standby.
- Program Scratch Pad Ram (PSPR) - Single cycle memory for local code fetch, data access via bus.
- Program Cache (PCache) - Single cycle memory for code fetch
- Local PFlash Bank (LPB) - Multi cycle flash memory, Local code and data read operations are direct access but may optionally be performed via the bus (see FLASHCON4)

## CPU Subsystem

### 5.3.6.1 Memory Addressing

This chapter details the CPU specific addressing.

#### 5.3.6.1.1 Local and Global Addressing

The TriCore architecture supports closely coupled program and data SRAM memories known as Program Scratch Pad RAM (PSPR) and Data Scratch Pad RAM (DSPR). The local PSPR memory is always located at C0000000H. The local DSPR is always located at D0000000H. In a multiprocessor system the local scratch pad memories appear in the global address map in the following locations:-

**Table 123 Global Address Locations**

CORE_ID Value	PSPR_base	DSPR_base
0	70100000 <sub>H</sub>	70000000 <sub>H</sub>
1	60100000 <sub>H</sub>	60000000 <sub>H</sub>
2	50100000 <sub>H</sub>	50000000 <sub>H</sub>
3	40100000 <sub>H</sub>	40000000 <sub>H</sub>
4	30100000 <sub>H</sub>	30000000 <sub>H</sub>
6	10100000 <sub>H</sub>	10000000 <sub>H</sub>

The CPUs always use the global addresses for bus transactions. Thus a data load from C0000000<sub>H</sub> (a CPUs local PSPR) will result in a bus transaction with an address in the range 10100000<sub>H</sub> - 701FFFFF<sub>H</sub> dependent on the Core-ID value of the processor. Similarly a code fetch form D0000000H (a CPUs local DSPR) will result in a bus transaction with an address in the range 10000000<sub>H</sub> - 700FFFFF<sub>H</sub> dependent on the Core-ID value of the processor.

#### 5.3.6.1.2 CSFR and SFR base Locations

Each CPU has a dedicated set of control and status registers accessed at the addresses detailed in the following table. These registers are divided into Special Function Registers (SFRs) and Core Special Function Registers (CSFRs).

A CPU must access its own CSFR registers using MTCR and MFCR instructions. CSFR registers of other CPUs may be accessed using load and store instructions via the XBAR\_SRI. SFR registers of any CPU may only be accessed using load and store instructions via XBAR\_SRI. Currently the overlay control and the access protection registers of CPUx are mapped into CPUx SFR address range.

The base locations for the TC1.6.2P SFR and CSFR registers are as follows:-

**Table 124 CSFR and SFR Base Locations**

CORE_ID Value	CSFR Base Address	SFR Base Address
0	F881_0000 <sub>H</sub>	F880_0000 <sub>H</sub>
1	F883_0000 <sub>H</sub>	F882_0000 <sub>H</sub>
2	F885_0000 <sub>H</sub>	F884_0000 <sub>H</sub>
3	F887_0000 <sub>H</sub>	F886_0000 <sub>H</sub>
4	F889_0000 <sub>H</sub>	F888_0000 <sub>H</sub>
6	F88D_0000 <sub>H</sub>	F88C_0000 <sub>H</sub>

**CPU Subsystem****5.3.6.1.3 Cache Memory Access**

The cache and tag memories may be mapped into the CPUs address space. When mapped the cache memories are contiguous with the DSPR/PSPR memories as detailed in the following table. When mapped the cache memories behave identically to the PSPR/DSPR memories and may be used as standard memory.

The mapping of cache and tag memories to the TriCore address space is controlled by the MTU\_MEMMAP register. See the MTU chapter for details.

**Table 125 Cache Memory Locations when mapped**

<b>Memory</b>	<b>Local Address</b>	<b>Global Address</b>
Program Cache	C000_0000 <sub>H</sub> + PSPR_Memory_Size	PSPR_Base + PSPR_Memory_Size
Data Cache	D000_0000 <sub>H</sub> + DSPR_Memory_Size	DSPR_Base + DSPR_Memory_Size

When mapped the tag memories are available at the locations detailed in the following table. The mapping of the Tag memories is provided test purposes only. They may not be used as standard memory.

**Table 126 Tag Memory Locations when mapped**

<b>Memory</b>	<b>Local Address</b>	<b>Global Address</b>
Program Tag	C00C_0000 <sub>H</sub>	PSPR_Base + 000C_0000 <sub>H</sub>
Data Tag	D00C_0000 <sub>H</sub>	DSPR_Base + 000C_0000 <sub>H</sub>

The CACHEI.\* instructions require a way and index value to be supplied in a valid address. (i.e. an address that will pass memory protection and null pointer checks). The CACHEI.\* instructions also require an address to be cacheable. The local portion of DLMU (even when accessed via segment 0x9) is not cacheable. The local DSPR's global address (segment 0x[7-core\_id] is not cacheable either). The location of these bits in the 32-bit address is as follows.

**Table 127 Way and Index Location**

<b>Function</b>	<b>Address Bits</b>
Way	[0]
Index	[12:5]

**5.3.6.1.4 Customer-ID Numbering**

In circumstances where the Infineon defined Core identification numbering scheme is insufficient or incompatible with a customer numbering scheme, a customer ID value may be supplied via the CUS\_ID register. The implemented numbering scheme used in early device steps did not scale well across the derivatives and was changed for the latest steps. This old numbering scheme used a reverse encoding and is limited to only a few steps listed below. Affected devices can be identified by reading the version number at address 0xAF40\_0CB8.

- Example: the value 0x10 01 03 00 read from 0xAF40\_0CB8 translates to V1.0.1.3.0.

**Table 128 Customer ID old numbering scheme**

		<b>TC39xAA<sup>1)</sup>, BA<sup>2)</sup></b>	<b>TC38xAA<sup>1)</sup>, AB<sup>3)</sup></b>	<b>TC35xAA<sup>4)</sup></b>
<b>Core</b>	<b>CORE_ID</b>	<b>CUS_ID</b>	<b>CUS_ID</b>	<b>CUS_ID</b>
CPU0	0	5	3	2
CPU1	1	4	2	1

## CPU Subsystem

**Table 128 Customer ID old numbering scheme (cont'd)**

		<b>TC39xAA<sup>1)</sup>, BA<sup>2)</sup></b>	<b>TC38xAA<sup>1)</sup>, AB<sup>3)</sup></b>	<b>TC35xAA<sup>4)</sup></b>
<b>Core</b>	<b>CORE_ID</b>	<b>CUS_ID</b>	<b>CUS_ID</b>	<b>CUS_ID</b>
CPU2	2	3	1	0
CPU3	3	2	0	
CPU4	4	1		
CPU5	6	0		

- 1) TC39xAA/TC38xAA: All devices use old numbering scheme
- 2) TC39xBA: Version above V1.0.1.3.0 use the new numbering scheme
- 3) TC38xAB: Version above V1.0.1.2.1 use the new numbering scheme
- 4) TC35xAA: Version above V1.0.0.1.2 use the new numbering scheme

Further design steps use the latest numbering scheme documented in the table below.

**Table 129 Customer ID numbering**

		<b>TC39xB</b>	<b>TC3Ex TC38x TC3Ax</b>	<b>TC37xEXT TC37x TC35x</b>	<b>TC36x TC33xEXT</b>	<b>TC33x</b>
<b>Core</b>	<b>CORE_ID</b>	<b>CUS_ID</b>	<b>CUS_ID</b>	<b>CUS_ID</b>	<b>CUS_ID</b>	<b>CUS_ID</b>
CPU0	0	0	0	0	0	0
CPU1	1	1	1	1	1	
CPU2	2	2	2	2		
CPU3	3	3	3			
CPU4	4	4				
CPU5	6	5				

### 5.3.6.2 Memory Integrity Error Handling

The TriCore CPUs contain integrated support for the detection and handling of memory integrity errors. The handling of memory integrity errors for the various memory types in the CPU is as follows:

#### 5.3.6.2.1 Program Side Memories

The program side memories of the CPU consist of four memory structures:- The Program Scratchpad RAM (PSPR), the Program Cache (PCACHE), the Program TAG RAM (PTAG) and the local PFlash bank (LPB). All of these memory structures are ECC protected from memory integrity errors. Any sub-width write access to the PSPR from the Bus interface is converted to a Read-Modify-Write sequence by the PMI module.

##### Program Scratchpad RAM (PSPR)

The Scratchpad RAM of the CPU is protected from memory integrity errors on a 64bit basis. ECC protection of the PSPR is enabled via the SSH ECCS register.

For instruction fetch requests from the TriCore CPU to PSPR, the ECC bits are read along with the data bits and are passed to the CPU along with their corresponding instructions. Whenever an attempt is made to issue an instruction containing an uncorrectable memory integrity error a synchronous PIE trap is raised. The trap handler is then responsible for correcting the memory entry and re-starting program execution.

## CPU Subsystem

For PSPR read operations from the Bus interface, either from the DMI module or another Bus master agent, an access that results in the detection of an uncorrectable memory integrity error in the requested data causes an error to be returned for the bus transaction. Since the TriCore CPU may not be involved in the transaction, a separate error alarm is also flagged to the SMU module to optionally generate an NMI trap back to the CPU.

Writes to program scratchpad memory are only ever performed from the bus interface. For write operations less than the protection width of the PSPR the memory transaction is transformed into a read-modify-write sequence inside the PMI module. Such a write operation may result in the detection of an uncorrectable memory integrity errors during the read phase which are handled as standard read operations.

### Program Cache (PCACHE)

The program cache RAM of the CPU is protected from memory integrity errors on a 64bit basis. ECC protection of the PCACHE is enabled via the SSH ECCS register.

For instruction fetch requests from the TriCore CPU to PCACHE, the ECC bits are read along with the data bits of all cache ways, and an uncorrectable error signal generated for each cache way. In the case of a tag hit, the uncorrectable error signals for the corresponding cache way are passed to the CPU along with their corresponding instructions. Whenever an attempt is made to issue an instruction containing an uncorrectable error a synchronous PIE trap is raised. The trap handler is then responsible for checking the source of the memory integrity error.

### Program Tag (PTag)

ECC protection of the PTAG is enabled via the SSH ECCS register.

For instruction fetch requests from the TriCore CPU to PCACHE, the program tag ECC bits are read along with the data bits and an error flag is computed. A way hit is triggered only if the tag address comparison succeeds, the valid bit is set and no ECC error in the associated tag way is detected, any other result is considered a miss. In the normal case where no error is detected in either cache way then the cache line is filled/refilled as normal. In the case where an error is detected the cache controller replacement algorithm forces the way indicating an error to be replaced. In the case where one cache way flags a cache hit, and another cache way detects an uncorrectable ECC error, the error condition is masked and has no effect on the memory integrity error handling mechanisms.

### Local Pflash Bank (LPB)

The local Pflash bank returns data protected on a 64bit basis (This is different to the protection of the actually flash memory array itself). For instruction fetch requests from the TriCore CPU to LPB, the ECC bits are read along with the data bits and are passed to the CPU along with their corresponding instructions. Whenever an attempt is made to issue an instruction containing an uncorrectable memory integrity error a synchronous PIE trap is raised.

For LPB read operations from the Bus interface an access that results in the detection of an uncorrectable memory integrity error in the requested data causes a bus error to be returned for the bus transaction. Since the TriCore CPU may not be involved in the transaction, a separate error is also flagged to the SMU module to optionally generate an NMI trap back to the CPU.

### 5.3.6.2.2 Data Side Memories

The program side memories of the CPU consist of four memory structures:- The Data Scratchpad RAM (DSPR), the Data Cache (DCACHE), The data TAG RAM and the Distributed LMU (DLMU). All of these memory structures are ECC protected from memory integrity errors. Any byte write access to either DSPR or DCache is converted to a half-word Read-Modify-Write sequence. Any sub-double-word write to the DLMU is converted into a read-modify-write operation. In normal operation isolated write transactions to the data memories result in no additional stall cycles.

## CPU Subsystem

### Data Scratchpad Ram (DSPR)

The DSPR memory of the TC1.6.2P is protected from memory integrity errors on a per half-word basis. ECC protection of the DSPR is enabled via the SSH ECCS register.

For data load requests from the TriCore CPU to DSPR, the ECC bits are read along with the data bits and an uncorrectable error signal is generated for each half-word. If an error is detected associated with any of the data half-words passed to the CPU an error is flagged to the CPU. If such an error condition is detected an asynchronous DIE trap is raised. The trap handler is then responsible for correcting the memory entry, or for taking alternative action (such as system soft reset) if correction of the data is not possible.

For DSPR read operations from the Bus interface, either from the PMI module or another Bus master agent, an access that results in the detection of an uncorrectable error in the requested data half-words causes an error to be returned for the bus transaction. Since the TriCore CPU may not be involved in the transaction, a separate error is also flagged to the SMU module to optionally generate an NMI trap back to the CPU.

For write operations to DSPR of half-word size or greater, the ECC bits are pre-calculated and written to the memory in parallel with the data bits. For byte write operations the memory transaction is transformed into a half-word read-modify-write sequence inside the DMI module. As such, byte write operations may result in the detection of uncorrectable memory integrity errors, which are handled as per standard read operations.

### Data Cache (DCache)

ECC protection of the DCACHE is enabled via the SSH ECCS register.

For data load requests from the TriCore CPU to DCache, the ECC bits are read along with the data bits of both cache ways, and an uncorrectable error flag computed for each half-word of each cache way. In the case where an error is detected with any of the requested data half-words in a cache way which has a corresponding tag hit, an error is flagged to the CPU. If such an error condition is detected an asynchronous DIE trap is raised. The trap handler is then responsible for correcting the memory entry, or for taking alternative action (such as system soft reset) if correction of the data is not possible.

For write operations of half-word size or greater, the check bits are pre-calculated and written to the memory in parallel with the data bits. For byte write operations the memory transaction is transformed into a half-word read-modify-write sequence inside the DMI module. As such, byte write operations may result in the detection of uncorrectable memory integrity errors as for read operations.

For cache line writeback, uncorrectable error detection is performed as dirty data is transferred to the store buffers. In all cases (normal cache line eviction, cachex.xx instruction) where an error condition is detected in a valid cache line a DIE trap is raised. The trap handler is then responsible for taking corrective action (such as system soft reset) since correction of the data is not possible.

### Data Tag (DTag)

ECC protection of the DTAG is enabled via the SSH ECCS register.

For data load or store requests from the TriCore CPU to DCache, the data tag ECC bits are read along with the data bits and an uncorrectable error flag is computed. A way hit is triggered only if the tag address comparison succeeds, the tag location is valid and no uncorrectable error in the associated tag way is detected, any other result is considered a miss. In the normal case where no error is detected in either tag way then the cache line is filled/refilled as normal. In the case of a cache miss where an error is detected in one of the tag ways and the cache line does not contain dirty data the cache controller replacement algorithm forces the way indicating an error to be replaced when the refill operation returns. In the case where one cache way flags a cache hit, and the another way detects an uncorrectable error, the error condition is masked and has no effect on the memory integrity error handling mechanisms. If a cache miss occurs, with an uncorrectable error detected on the associated data tag way and dirty data detected, then an asynchronous DIE trap is signalled to the CPU and any writeback / refill sequence aborted. The trap handler is responsible for invalidating the cache line and processing any associated

---

## CPU Subsystem

dirty data if possible, or taking other corrective action. Similar action is taken for forced cache writeback using the cache manipulation instructions.

### Distributed LMU (DLMU)

The DLMU memory of the TC1.6.2P is protected from memory integrity errors on a double word (64bit) basis. ECC protection of the DLMU is enabled via the SSH\_ECCS register.

For data load requests from the TriCore CPU to DLMU, the ECC bits are read along with the data bits and an uncorrectable error signal is generated for each double word. If an error is detected associated with any of the data double-words passed to the CPU an error is flagged to the CPU. If such an error condition is detected an asynchronous DIE trap is raised. The trap handler is then responsible for correcting the memory entry, or for taking alternative action (such as system soft reset) if correction of the data is not possible.

For DLMU read operations from the Bus interface, either from the PMI module or another Bus master agent, an access that results in the detection of an uncorrectable error in the requested data double-words causes a bus error to be returned for the bus transaction. Since the TriCore CPU may not be involved in the transaction, a separate alarm is also flagged to the SMU module.

For write operations to DLMU of double-word size or greater, the ECC bits are pre-calculated and written to the memory in parallel with the data bits. For sub double word write operations the memory transaction is transformed into a read-modify-write sequence. Such operations may result in the detection of uncorrectable memory integrity errors, which are handled as per standard read operations.

#### 5.3.6.2.3 Memory Initialisation

To avoid the generation of spurious ECC errors the DLMU, DSPR, PSPR, LPB must be fully initialised prior to use. This may be done either by software or by automatically by hardware (see the DMU.HF\_PROCONRAM register for details). The entire physical memory as detailed in [Chapter 5.3.3](#) must be initialised irrespective of any memory size variants produced for derivative products.

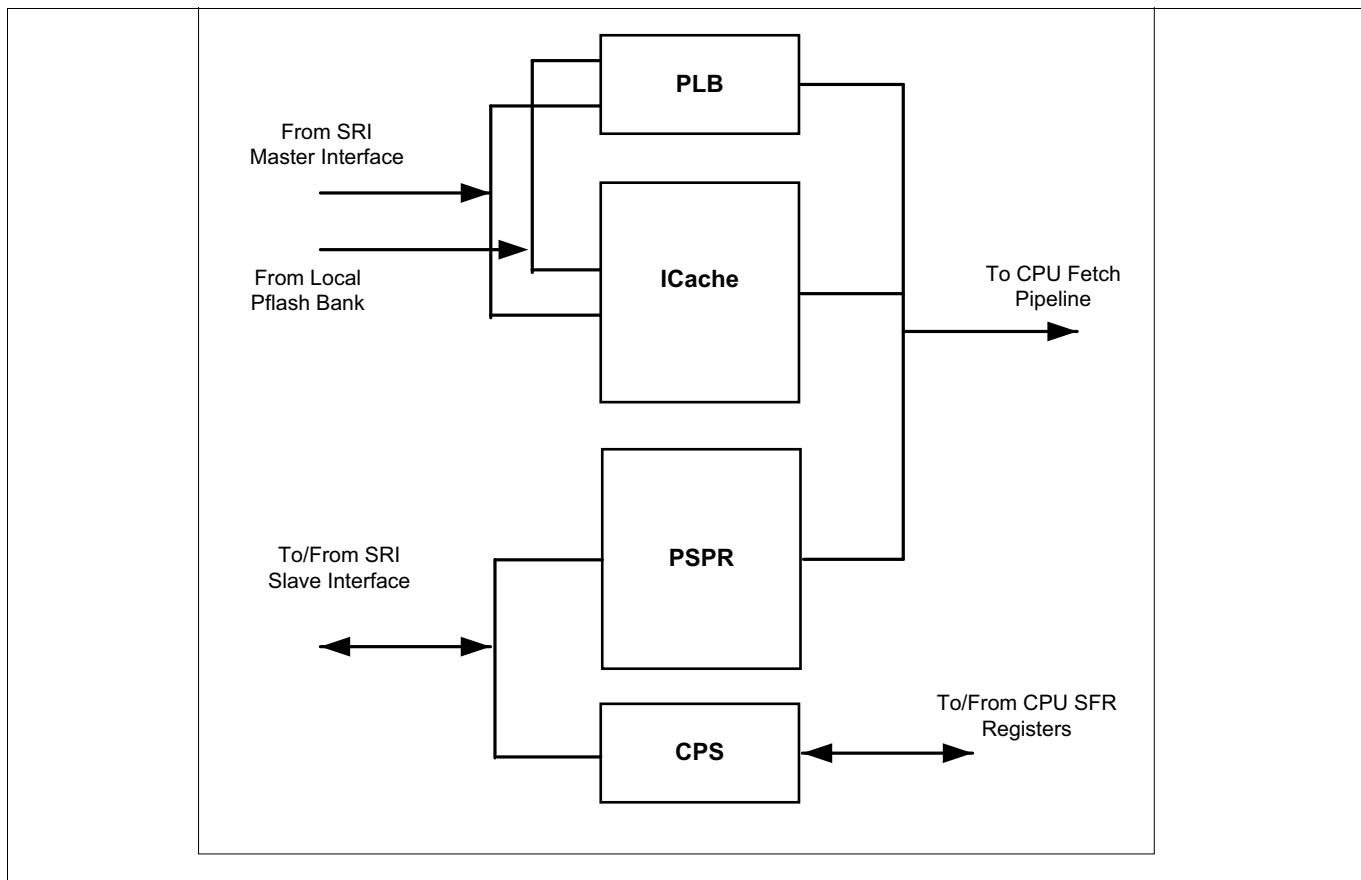
## CPU Subsystem

### 5.3.6.3 Program Memory Interface (PMI)

The program Memory Interface (PMI) provides the instruction stream to the CPU.

#### 5.3.6.3.1 TC1.6.2P PMI Description

**Figure 51** shows the block diagram of the Program Memory Interface (PMI) of the TC1.6.2P.



**Figure 51 PMI Block Diagram**

The Program Memory Interface (PMI) has the following features:

- Program Cache (PCACHE)
  - Two-way set associative cache
  - LRU (Least-Recently Used) replacement algorithm
  - Cache line size: 256 bits (4 double-words)
  - Validity granularity: One valid bit per cache line
  - Single cycle access for hit.
  - The PCACHE can be globally invalidated to provide support for software cache coherency (to be handled by the programmer)
  - The PCACHE can be bypassed to provide a direct fetch from the CPU to on-chip and off-chip resources
  - The PCACHE refill mechanism is critical word first with line wrap around, and direct streaming to the CPU
- Program Scratchpad memory (PSPR)
- CPU interface
  - Supporting 64bit aligned fetches.

## CPU Subsystem

- CPU Slave interface (CPS)
- Shared Resource Interconnect Bus (SRI) Master Interface
- Interface to the local PFlash Bank (LPB)
- Shared Resource Interconnect Bus (SRI) Slave Interface to PSPR and CPS.
- All PMI SRAMs (PSPR, PCACHE, and cache tag SRAM) are ECC protected
  - ECC is calculated on 64bit data for the PSPR and PCACHE

### Program Scratchpad RAM (PSPR)

The TC1.6.2P Program scratchpad RAM provides a fast, deterministic program fetch access from the CPU for use by performance critical code sequences.

- CPU program fetch accesses to scratchpad RAM are never cached in the program cache and are always directly targeted to the scratchpad RAM.

The CPU fetch interface will generate aligned accesses (64-bit), which will result in 64-bits of instruction being returned to the CPU.

Note that the CPU Fetch Unit can only read from the scratchpad RAM and can never write to it.

The scratchpad RAM may also be accessed from the SRI Slave interface by another bus master, such as the Data Memory Interface (DMI). The scratchpad RAM may be both read and written from the SRI. The SRI Slave interface supports all SRI transaction types.

The Program scratchpad RAM is ECC protected across 64-bit data. Writes to the PSPR that are less than 64 bits, or which are 64 bits but not naturally aligned will be performed as read-modify-write operations at the memory interface. Any read-modify-write operation that detects an error on the read phase will not perform the write phase of operation. (Sub-64 bit writes to un-initialised memory will always result in an ECC error being detected on the read phase and hence no data will be written to the memory).

### Program Cache

The program Cache is a two-way set-associative cache with a Least-Recently-Used (LRU) replacement algorithm. Each PCACHE line contains 256 bits of instruction and associated ECC bits.

CPU program fetch accesses which target a cacheable memory segment (and where the PCACHE is not bypassed) target the PCACHE. If the requested address and its associated instruction are found in the cache (Cache Hit), the instruction is passed to the CPU Fetch Unit without incurring any wait states. If the address is not found in the cache (Cache Miss), the PMI cache controller issues a cache refill sequence and wait states are incurred whilst the cache line is refilled. The fetch request always returns an aligned packet to the CPU.

The program cache is ECC protected on 64-bit data.

The program TAG Ram is ECC protected on 22-bit data.

Errors detected at the ECC decoders are signaled to the EMM via the SSH logic of the associated SRAM

### Program Cache Refill Sequence

Program Cache refills are performed using a critical double-word first strategy with cache line wrapping such that the refill size is always 4 double-words. PCACHE refills are always performed in 64-bit quantities. A refill sequence will always affect only one cache line. There is no prefetching of the next cache line.

PCACHE refills are therefore implemented using an SRI Burst Transfer 4 (BTR4) transfers. The program cache supports instruction streaming, meaning that it can deliver available instruction half-words to the CPU Fetch Unit whilst the refill operation is ongoing.

## CPU Subsystem

### Program Cache Bypass

The Program Cache may be bypassed, under control of PCON0.PCBYP, to provide a direct instruction fetch path for the CPU Fetch Unit. The default value of PCON0.PCBYP is such that the PCACHE is bypassed after reset.

Whilst PCACHE bypass is enabled, a fetch request by the CPU to a cacheable address will result in a forced cache miss, such that the cache controller issues a standard refill sequence and supplies instruction half-words to the CPU using instruction streaming, without updating the cache contents. Any valid cache lines within the PCACHE will remain valid and unchanged whilst the PCACHE is bypassed. As such, instruction fetch requests to cacheable addresses with PCACHE bypass enabled behave identically to instruction fetch requests to non-cacheable addresses.

The PCON0 register is CPUx endinit protected.

### Program Cache Invalidation

The PMI does not have automatic cache coherency support. Changes to the contents of memory areas external to the PMI that may have already been cached in the PCACHE are not detected. Software must provide the cache coherency in such a case. The PMI supports this via the cache invalidation function. The PCACHE contents may be globally invalidated by writing a '1' to PCON1.PCINV. The PCACHE invalidation is performed over 64 cycles by a hardware state machine which cycles through the PCACHE entries marking each as invalid. During an invalidate sequence the CPU may continue to fetch instructions from non-cacheable memory. Any attempt to fetch instructions from a cacheable memory location during an invalidation sequence will result in the CPU stalling until the sequence completes. The status of the PCACHE invalidation sequence may be determined by reading the PCON1.PCINV bit.

### Program Line Buffer (PLB)

The PMI module contains a 256-bit Program Line Buffer (PLB). Program fetch requests to non-cacheable addresses (or to cacheable addresses with the cache in bypass) utilize the PLB as a single line cache. A single valid bit is associated with the PLB, denoting that the PLB contents are valid. As such all fetch requests resulting in an update of the PLB, whether to a cacheable address or not, are implemented as SRI Burst Transfer 4 (BTR4) transactions, with the critical double-word of the PLB line being fetched first size. The PLB may be invalidated by writing PCON1.PBINV.

### CPU Slave Interface (CPS)

The CPU Slave Interface provides access from the SRI bus to the CPU CSFR and SFR registers.

## CPU Subsystem

### 5.3.6.3.2 PMI Registers

Three control registers are control the operation of the Program Memory Interface. These registers and their bits are described in this section.

#### PMI Register Descriptions

##### CPUx Program Control 0

###### PCON0

###### CPUx Program Control 0

(1920C<sub>H</sub>)

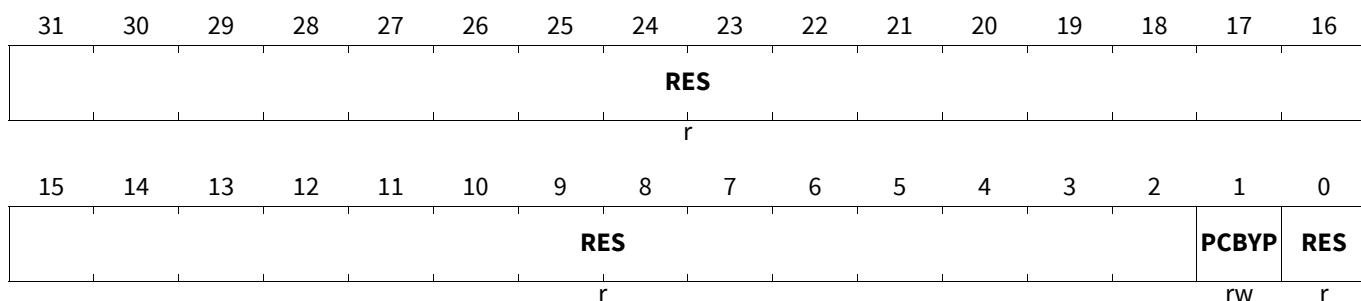
**Application Reset Value:** 0000 0002<sub>H</sub>

###### CPU\_PCON0

###### Short address for domain CSFR

(0920C<sub>H</sub>)

**Application Reset Value:** 0000 0002<sub>H</sub>



Field	Bits	Type	Description
<b>RES</b>	0, 31:2	r	<b>Reserved</b> Read as 0; should be written with 0.
<b>PCBYP</b>	1	rw	<b>Program Cache Bypass</b> 0 <sub>B</sub> Cache enabled 1 <sub>B</sub> Cache bypass (disabled)

##### CPUx Program Control 1

###### PCON1

###### CPUx Program Control 1

(19204<sub>H</sub>)

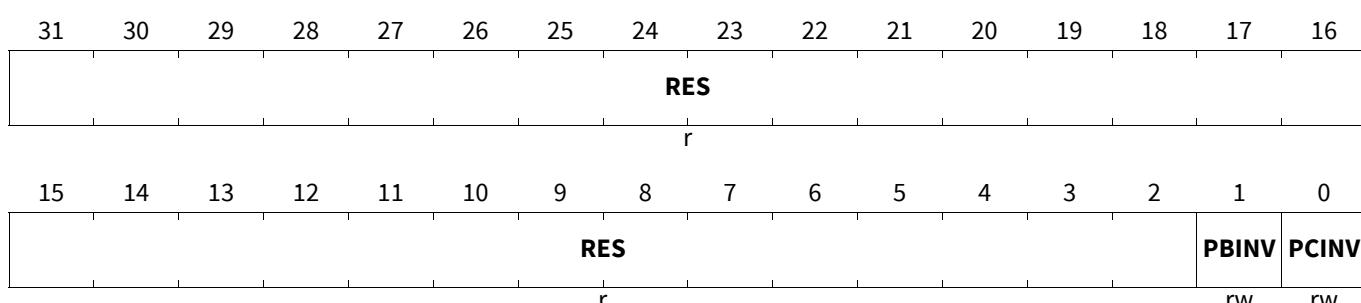
**Application Reset Value:** 0000 0000<sub>H</sub>

###### CPU\_PCON1

###### Short address for domain CSFR

(09204<sub>H</sub>)

**Application Reset Value:** 0000 0000<sub>H</sub>



## CPU Subsystem

Field	Bits	Type	Description
PCINV	0	rw	<b>Program Cache Invalidate</b> 0 <sub>B</sub> Write: No effect, normal instruction cache operation. Read : Normal operation, instruction cache available 1 <sub>B</sub> Write : Initiate invalidation of entire instruction cache. Read: Instruction cache invalidation in progress. Instruction cache unavailable.
PBINV	1	rw	<b>Program Buffer Invalidate</b> Write Operation: This field returns 0 when read. 0 <sub>B</sub> Write: No effect. Normal program line buffer operation. 1 <sub>B</sub> Write :Invalidate the program line buffer.
RES	31:2	r	<b>Reserved</b> Read as 0; should be written with 0.

## CPUx Program Control 2

### PCON2

#### CPUx Program Control 2

(19208<sub>H</sub>)

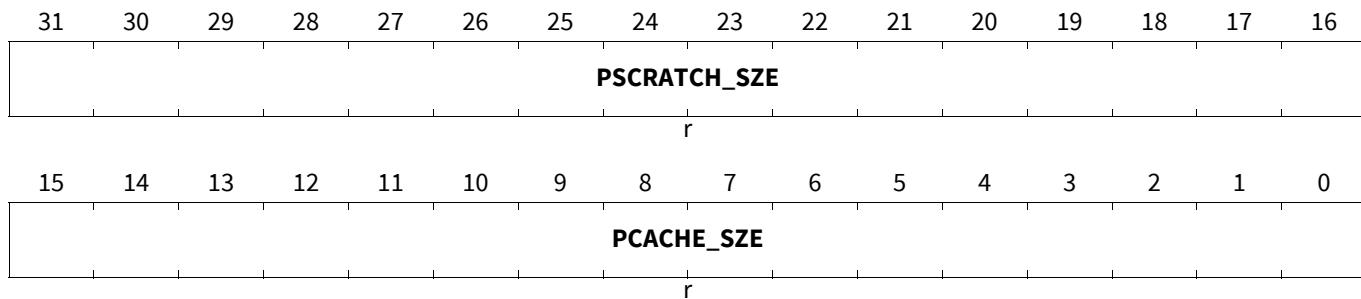
Application Reset Value: XXXX XXXX<sub>H</sub>

#### CPU\_PCON2

#### Short address for domain CSFR

(09208<sub>H</sub>)

Application Reset Value: XXXX XXXX<sub>H</sub>



Field	Bits	Type	Description
PCACHE_SZE	15:0	r	<b>Program Cache Size (ICACHE) in KBytes</b> In KBytes
PSCRATCH_SZE	31:16	r	<b>Program Scratch Size in KBytes</b> In KBytes

## CPUx Program Synchronous Trap Register

PSTR contains synchronous trap information for the program memory system. The register is updated with trap information for PSE traps to aid the localisation of faults. The register is only set whenever a trap is detected and the register has no bits already set. It is cleared by a CSFR write (independent of data value).

## CPU Subsystem

### PSTR

**CPUx Program Synchronous Trap Register** (19200<sub>H</sub>)

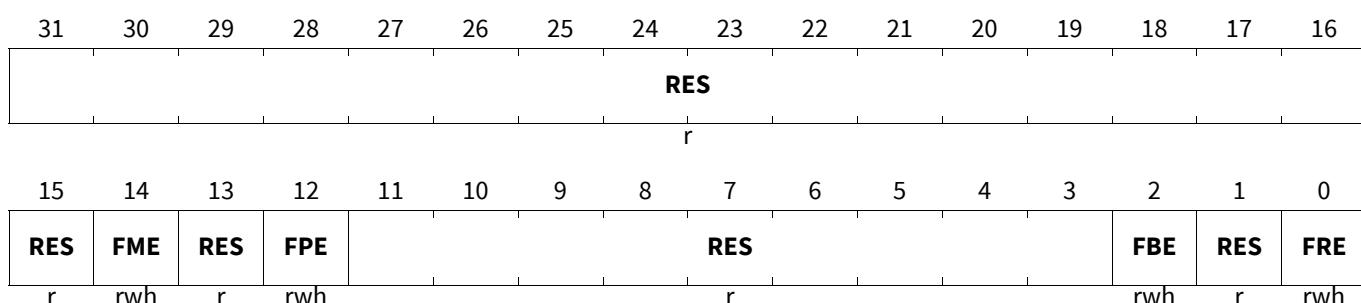
**Application Reset Value:** 0000 0000<sub>H</sub>

**CPU\_PSTR**

**Short address for domain CSFR**

(09200<sub>H</sub>)

**Application Reset Value:** 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>FRE</b>	0	rwh	<b>Fetch Range Error</b> A Fetch Range Error occurs whenever an access to the Program Scratch is outside the range of the SRAM.
<b>RES</b>	1, 11:3, 13, 31:15	r	<b>Reserved</b>
<b>FBE</b>	2	rwh	<b>Fetch Bus Error</b> A Fetch bus error will be set whenever the SRI flags an error due a fetch from external memory. This will be set for both direct fetches from the bus and for cache refills.
<b>FPE</b>	12	rwh	<b>Fetch Peripheral Error</b> A Fetch peripheral error will be flagged whenever a fetch is attempted to peripheral space.
<b>FME</b>	14	rwh	<b>Fetch MSIST Error</b> During SIST mode, a fetch from the PTAG will cause a PSE trap to occur.

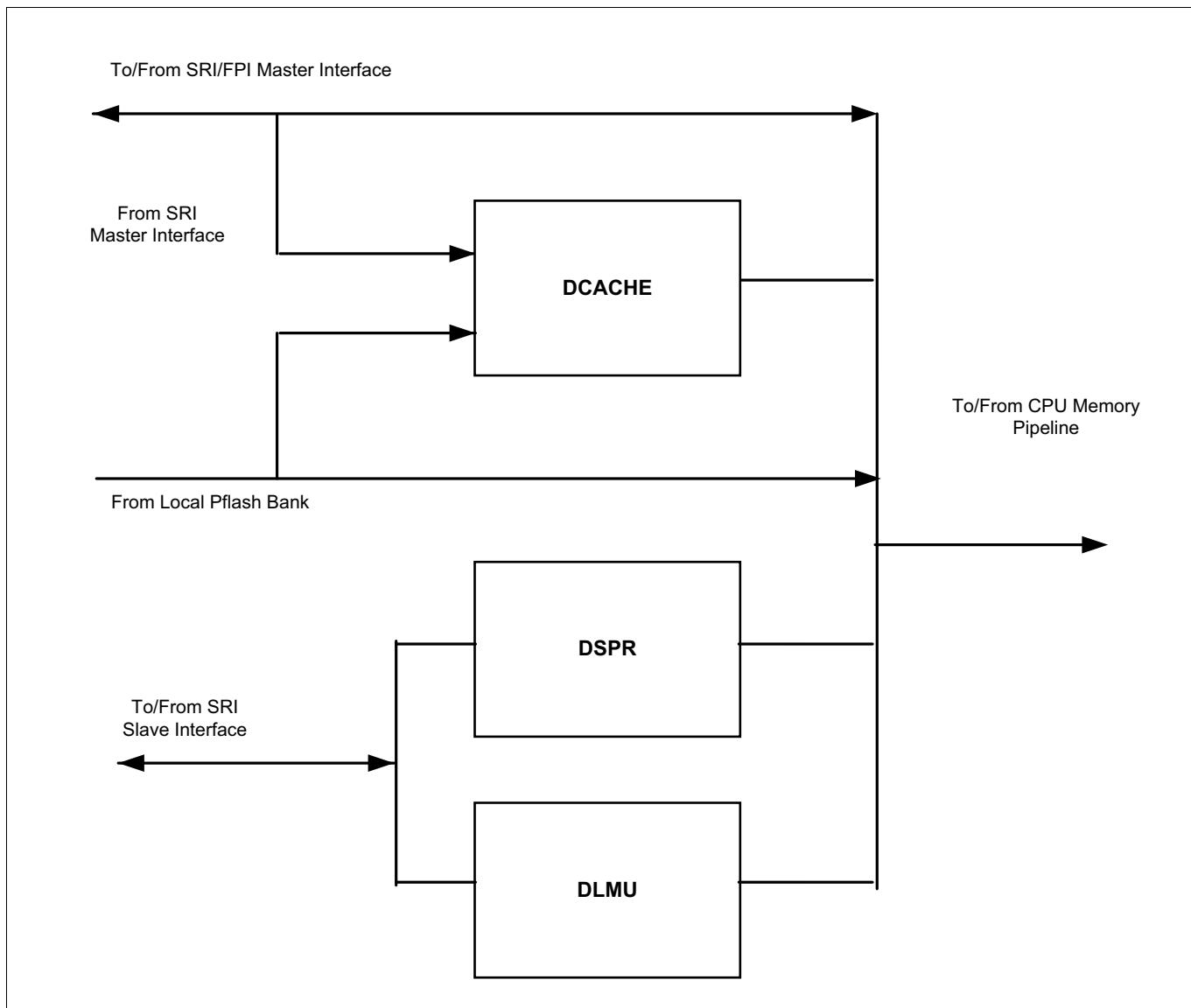
## CPU Subsystem

### 5.3.6.4 Data Memory Interface (DMI)

The Data Memory Interface (DMI) provides data values to the CPU and stores data values supplied by the CPU.

#### 5.3.6.4.1 DMI Description

This figure shows the block diagram of the Data Memory Interface (DMI) of the TC1.6.2P.



**Figure 52 DMI block diagram**

The Data Memory Interface (DMI) has the following features:

- Data Scratchpad Ram (DSPR)
  - Supporting unaligned access (16-bit aligned) with no penalty.
- Data Memory (DCACHE):
  - Two-way set associative cache, Least recently used (LRU) replacement algorithm
  - Cache line size: 256 bits
  - Validity granularity: One valid bit per cache line
  - Write-back Cache: Writeback granularity: 256 bits

## CPU Subsystem

- Refill mechanism: full cache line refill
- Single cycle access for hit.
- Distributed LMU SRAM (DLMU)
- CPU interface
- Shared Resource Interconnect Bus (SRI) Master interface
- Shared Resource Interconnect (SRI) Slave interface to DSPR
- Interface to local PFlash bank (LPB)
- Flexible Peripheral Interconnect Bus (FPI) Master interface for fast access to peripherals
- All DMI SRAMs (DSPR, DCACHE, and cache tag SRAM) are ECC protected

### Data Scratchpad RAM (DSPR)

The DSPR provides fast, deterministic data access to the CPU for use by performance critical code sequences.

The DSPR is organised as multiple memory “towers”. This organisation allow the CPU to access 64bits of data from any 16bit aligned address

The DSPR may also be accessed from the SRI Slave interface by another bus master, with both read and write transactions supported. The DSPR may be accessed by the SRI Slave interface using any SRI transaction type, including burst transfers. In accordance with the SRI protocol, accesses to the SRI Slave interface must be naturally aligned.

Errors detected at the ECC decoders are signaled to the EMM via the SSH logic of the associated SRAM.

The data scratchpad RAM is ECC protected across 16-bit data. Bytes Writes to the DSPR will be performed as read-modify-write operations at the memory interface. Any read-modify-write operation that detects an error on the read phase will not perform the write phase of operation. (Byte writes to un-initialised memory will always result in an ECC error being detected on the read phase and hence no data will be written to the memory).

### Data Cache (DCACHE)

The DCache is a Two-way set-associative cache with a Least-Recently-Used (LRU) replacement algorithm. Each line contains 256 bits of data along with ECC bits. A single valid bit and a single dirty bit are associated with each line.

CPU data accesses to a cacheable memory segment target the DCache. If the requested address and its associated data are found in the cache (Cache Hit), the data is passed to/from the CPU Load-Store Unit without incurring any wait states. If the address is not found in the cache (Cache Miss), the DMI cache controller issues a cache refill sequence and wait states are incurred whilst the cache line is refilled. The CPU load-store interface will generate unaligned accesses (16-bit aligned), which will result in up to 64-bits of data being transferred to or from the CPU (for non-context operations). If the data access is made within a DCACHE line, no matter the alignment, and a cache hit is detected then the requested data is returned to the CPU in a single cycle. If the data access is made to the end of a DCACHE line, such that the requested data would span two DCACHE lines, a single wait cycle is incurred (if both cache lines are present in the cache, otherwise a refill sequence is required for the missing cache line(s)).

The data cache is of the writeback type. When the CPU writes to a cacheable location the data is merged with the corresponding cache line and not written to main memory immediately. Associated with each cache line is a single ‘dirty’ bit, to denote that the data in the cache line has been modified. Whenever a CPU load-store access results in a cache miss, and each of the potential cache ways that could hold the requested cache line are valid, one of the cache lines is chosen for eviction based upon the LRU replacement algorithm. The line selected for eviction is then checked to determine if it has been modified using its dirty bit. If the line has not been modified the line is discarded and the refill sequence started immediately. If the line has been modified then the dirty data

## CPU Subsystem

is first written back to main memory before the refill is initiated. Due to the single dirty bit per cache line, 256 bits of data will always be written back, resulting in a SRI Burst-4 Transfer (BTR4) transactions.

Data Cache refills always result in the full cache line being refilled, with the critical double-word of the DCache line being fetched first. A refill sequence will always affect only one cache line. There is no prefetching of the next cache line. Due to the uniform size of DCache refill sequences, such refills are always implemented using SRI Burst Transfer 4 (BTR4) transactions.

All Cache SRAMs are ECC protected.

All TAG SRAMs are ECC protected

Errors detected at the ECC decoders are signaled to the EMM via the SSH logic of the associated SRAM

### Data Cache Bypass

The Data Cache may be bypassed, under control of DCON0.DCBYP, to provide a direct data access path to memory. The default value of DCON0.DCBYP is such that the data cache is bypassed after reset.

Whilst the data cache bypass is enabled, a data access request by the CPU to a cacheable address will result in a forced cache miss. Any valid cache lines within the data cache will remain valid and unchanged whilst the data cache is bypassed. As such data accesses to cacheable addresses with the data cache bypass enabled behave identically to data accesses to non-cacheable addresses.

The DCON0 register is CPUx endinit protected.

### 5.3.6.4.2 Distributed LMU (DLMU)

The DLMU is a contiguous portion of the global LMU SRAM. The DLMU provides fast, deterministic data access to a segment of the global LMU, ideally for use by the local CPU for global data.

The DLMU is organised as a 64 bit wide memory with ECC protection implemented on a 64bit granularity. Sub double word write accesses are performed as read-modify-write operations.

The DLMU may also be accessed from the SRI Slave interface by another bus master, with both read and write transactions supported. The DSPR may be accessed by the SRI Slave interface using any SRI transaction type, including burst transfers.

**The DLMU is uncacheable for data accesses from the local CPU. (The same as the DSPR)**

Errors detected at the ECC decoders are signaled to the SMU via the SSH logic of the associated SRAM.

The DLMU is ECC protected across 64-bit data. Writes to the DLMU that are less than 64 bits, or which are 64 bits but not naturally aligned will be performed as read-modify-write operations at the memory interface. Any read-modify-write operation that detects an error on the read phase will not perform the write phase of operation. (Sub-64 bit writes to un-initialised memory will always result in an ECC error being detected on the read phase and hence no data will be written to the memory).

### 5.3.6.4.3 DMI Trap Generation

CPU data accesses to the DMI may encounter one of a number of potential error conditions, which result in one of the following trap conditions being reported by the DMI.

#### ALN Trap

An ALN trap is raised for the following conditions:

- An access whose effective address does not conform to the alignment rules
- An access where the length, size or index of a circular buffer is incorrect

Whenever an ALN trap occurs, the DSTR (Data Synchronous Trap Register) and the DEADD (Data Error Address Register) CSFRs are updated.

## CPU Subsystem

### MEM Trap

A MEM trap is raised for the following conditions:

- An access whose effective address has a different segment to that of the base address (Segment Difference Error)
- An access whose effective address causes the data to span two segments (Segment Crossing Error)
- A memory address is used to access a CSFR area (CSFR Access Error)

Whenever a MEM trap occurs, the DSTR (Data Synchronous Trap Register) and the DEADD (Data Error Address Register) CSFRs are updated.

### DSE Trap

A DSE trap is raised for the following conditions:

- An access outside the range of the DSPR (Scratch Range Error)
- An error on the bus for an external accesses due to a load (Load Bus Error)
- An error from the bus during a cache refill (Cache Refill Error)
- An error during a load whilst in SIST mode (Load MSIST Error)
- An error generated by the overlay system during a load.

Whenever a DSE trap occurs, the DSTR (Data Synchronous Trap Register) and the DEADD (Data Error Address Register) CSFRs are updated.

### DAE Trap

A DAE trap is raised for the following conditions:

- An error on the bus for an external accesses due to a store (Store Bus Error)
- An error on the bus due to a cache writeback (Cache writeback Error)
- An error from the bus due to a cache flush (Cache Flush Error)
- An error due to a store whilst in SIST mode (Store MSIST Error)
- An error generated by the overlay system during a store.

Whenever a non-inhibited DAE trap occurs, the DATR (Data Asynchronous Trap Register) and the DEADD (Data Error Address Register) CSFRs are updated. DAE traps are inhibited if the DATR register is non-zero.

### Data Memory Protection Traps

Data memory protection traps (MPW, MPR, MPP, MPN) are raised by the memory protection system when a protection violation occurs. Whenever a data memory protection trap occurs the DSTR (Data synchronous trap register) and the DEADD (Data Error Address Register) are updated.

### 5.3.6.4.4 DMI Registers

Two control registers and three trap flag registers control the operation of the DMI. These registers and their bits are described in this section.

## CPU Subsystem

### DMI Register Descriptions

Note: There is no DCON1 register in this implementation.

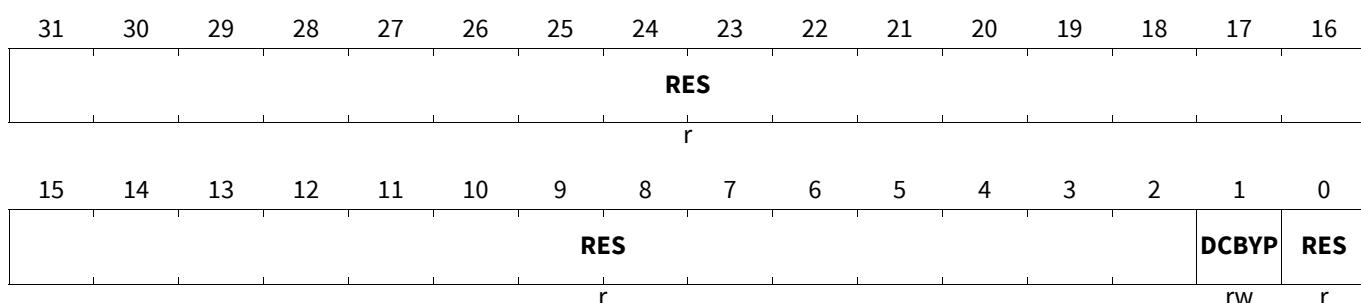
#### CPUx Data Memory Control Register

##### DCON0

**CPUx Data Memory Control Register** **(19040<sub>H</sub>)** **Application Reset Value: 0000 0002<sub>H</sub>**

##### CPU\_DCON0

**Short address for domain CSFR** **(09040<sub>H</sub>)** **Application Reset Value: 0000 0002<sub>H</sub>**



Field	Bits	Type	Description
RES	0, 31:2	r	Reserved
DCBYP	1	rw	<b>Data Cache Bypass</b> 0 <sub>B</sub> DCache / DRB enabled 1 <sub>B</sub> DCache / DRB Bypass (disabled)

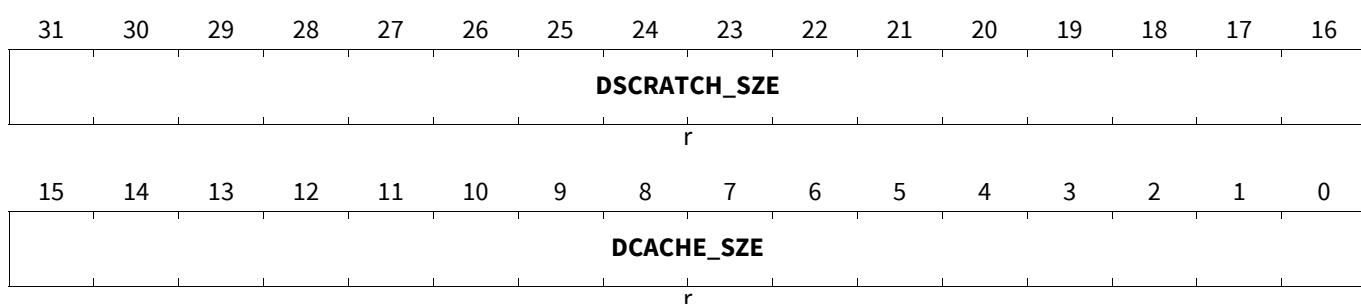
#### CPUx Data Control Register 2

##### DCON2

**CPUx Data Control Register 2** **(19000<sub>H</sub>)** **Application Reset Value: XXXX XXXX<sub>H</sub>**

##### CPU\_DCON2

**Short address for domain CSFR** **(09000<sub>H</sub>)** **Application Reset Value: XXXX XXXX<sub>H</sub>**



Field	Bits	Type	Description
DCACHE_SZE	15:0	r	<b>Data Cache Size</b> In KBytes
DSCRATCH_SZE	31:16	r	<b>Data Scratch Size</b> In KBytes

## CPU Subsystem

## **CPUx Data Synchronous Trap Register**

The DSTR contains synchronous trap information for the data memory system. The register is updated with trap source information to aid the localisation of faults.

The register is updated whenever a valid trap is detected and the register has no bits already set. It is cleared by a write (independent of data value).

DSTR

<b>CPUx Data Synchronous Trap Register</b>										<b>(19010H)</b>		<b>Application Reset Value: 0000 0000H</b>					
<b>CPU_DSTR</b>																	
<b>Short address for domain CSFR</b>										<b>(09010H)</b>		<b>Application Reset Value: 0000 0000H</b>					
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
RES										ALN	RES		CLE	MPE	CAC	SCE	SDE
r										rwh	r		rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
LOE	DTME	RES						CRE	RES		DRE	LBE	GAE	SRE			
rwh	rwh	r						rwh	r		rwh	rwh	rwh	rwh	rwh	rwh	

Field	Bits	Type	Description
SRE	0	rwh	<b>Scratch Range Error</b> A scratch Range Error occurs whenever an access to the data scratch is outside the range of the SRAM.
GAE	1	rwh	<b>Global Address Error</b> Load or store to local code scratch address outside of the lower 1MByte.
LBE	2	rwh	<b>Load Bus Error</b> A Load Bus Error will be set whenever the SRI flags an error due a load from external memory.
DRE	3	rwh	<b>Local DLMU Range Error</b> A DLMU Range Error occurs whenever an access to the local DLMU region is outside the physically implemented memory.
RES	5:4, 13:7, 23:21, 31:25	r	<b>Reserved</b>
CRE	6	rwh	<b>Cache Refill Error</b> A Cache Refill Error will be set whenever the SRI flags an error due a cache refill from external memory.
DTME	14	rwh	<b>DTAG MSIST Error</b> Access to memory mapped DTAG range outside of physically implemented memory.
LOE	15	rwh	<b>Load Overlay Error</b> Load to invalid overlay address.
SDE	16	rwh	<b>Segment Difference Error</b> Load or store access where base address is in different segment to access address.

## CPU Subsystem

Field	Bits	Type	Description
<b>SCE</b>	17	rwh	<b>Segment Crossing Error</b> Load or store access across segment boundary.
<b>CAC</b>	18	rwh	<b>CSFR Access Error</b> Load or store to local CSFR space.
<b>MPE</b>	19	rwh	<b>Memory Protection Error</b> Data access violating memory protection.
<b>CLE</b>	20	rwh	<b>Context Location Error</b> Context operation to invalid location.
<b>ALN</b>	24	rwh	<b>Alignment Error</b> Data access causing alignment error.

### CPUx Data Asynchronous Trap Register

The DATR contains asynchronous trap information for the data memory system. The register is updated with trap information for DAE traps to aid the localisation of faults.

The register is updated whenever a valid trap is detected and the register has no bits already set. It is cleared by a write (independent of data value).

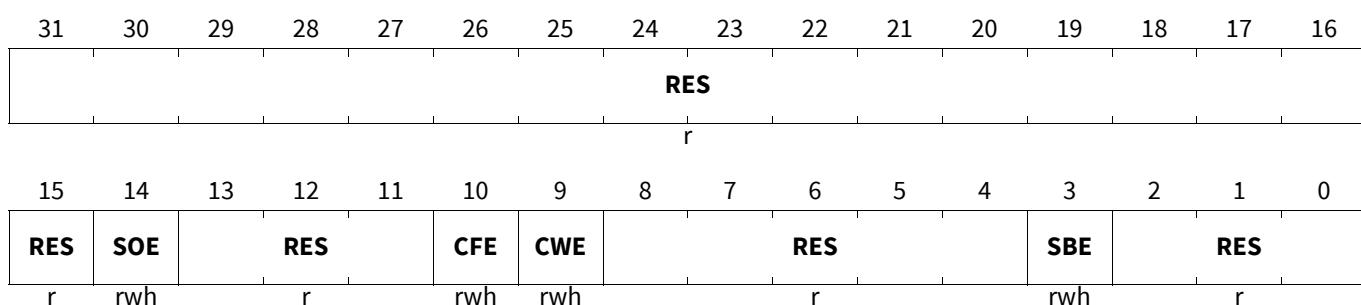
DAE traps are inhibited if the DATR register is non-zero.

#### DATR

**CPUx Data Asynchronous Trap Register** **(19018<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**

#### CPU\_DATR

**Short address for domain CSFR** **(09018<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>RES</b>	2:0, 8:4, 13:11, 15, 31:16	r	<b>Reserved</b>
<b>SBE</b>	3	rwh	<b>Store Bus Error</b>
<b>CWE</b>	9	rwh	<b>Cache Writeback Error</b>
<b>CFE</b>	10	rwh	<b>Cache Flush Error</b>
<b>SOE</b>	14	rwh	<b>Store Overlay Error</b>

## CPU Subsystem

### CPUx Data Error Address Register

DEADD contains trap address information for the Data memory system. The register is updated with trap information for MEM, ALN, DSE or DAE traps to aid the localisation of faults.

The register is only set whenever a trap is detected and either the DATR or DSTR registers have no bits already set.

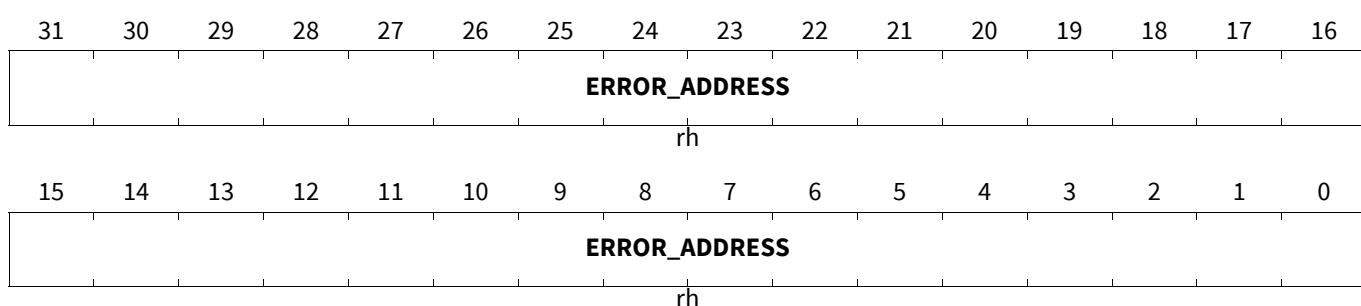
The register contents are only valid when either the DATR or DSTR register is non-zero and hence should be read prior to clearing these registers.

#### DEADD

**CPUx Data Error Address Register** **(1901C<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**

**CPU\_DEADD**

**Short address for domain CSFR** **(0901C<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>ERROR_ADDR</b>	31:0	rh	<b>Error Address</b>
<b>ESS</b>			

### 5.3.7 Miscellaneous

This chapter documents miscellaneous features that do not have any other logical place to be documented.

#### 5.3.7.1 Boot Halt

Following a system reset Core-0 will start its boot sequence, all other cores will be placed in the boot-halt state. In this state the core is halted, and the instruction fetch system stalled. All traps and interrupts are ignored by a core in the boot-halt state. All memories and registers associated with the core are available for access via the bus system in the boot-halt state. A core in boot-halt has SYSCON.BHALT set. **The core will remain halted until the SYSCON.BHALT bit is written to "0" when it will start to fetch from the PC defined in the PC CSFR (This will be the boot PC unless otherwise updated). A write to "1" of the SYSCON.BHALT will be ignored.**

#### 5.3.7.2 SSH usage recommendations

The following usage of the CPU SRAM SSH system is recommended.

- When any one of the CPU local data memories (DSPR/DCACHE or DTAG) is in SSH test mode (MEMTEST enabled), then other CPU local data memories will be not be accessible. It is therefore recommended to enable DSPR/DCACHE SSH mode and DTAG memory SSH mode together.
- When any one of the CPU local program memories (PSPR/PCACHE or PTAG) is in SSH test mode (MEMTEST enabled), then other CPU local program memories will be not be accessible. It is therefore recommended to enable PSPR/PCACHE SSH mode and PTAG memory SSH mode together.

---

## CPU Subsystem

### 5.3.7.3 Debug restrictions

For context operations and operations using circular addressing the base address of the operation is used for debug range comparison not the effective address.

When the pcache is mapped into address space pcache operation can only be enabled when debug is enabled. When the dcache is mapped into address space dcache operation can only be enabled when debug is enabled.

External writes to the processor's GPR registers are not supported unless the processor is in Boot-Halt, Debug-Halt or idle.

External writes to the processor's PSW and PC registers are not supported unless the processor is in Boot-Halt, Debug-Halt or idle.

### 5.3.7.4 Local Pflash Bank Configuration Registers

## CPU Subsystem

## 5.3.7.4.1 Registers

## Flash Configuration Registers

## CPUx Flash Configuration Register 0

Software may program a Flash Prefetch Buffer with a master tag identifier stored in Flash Configuration Register 0.

If a CPU instance does not have a local PFlash bank then the FLASHCON0 register associated with that instance will have no functionality.

## FLASHCON0

CPUx Flash Configuration Register 0 (01100<sub>H</sub>) Reset Value: [Table 130](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>RES</b>								<b>RES</b>							<b>TAG3</b>
r								r							rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>RES</b>								<b>RES</b>							<b>TAG1</b>
r								r							rw

Field	Bits	Type	Description
<b>TAG1</b>	5:0	rw	<b>Flash Prefetch Buffer 1 Configuration</b> FPB is assigned to on chip bus master with master tag id equal to TAG1.
<b>RES</b>	7:6, 15:14, 23:22, 31:30	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>TAG2</b>	13:8	rw	<b>Flash Prefetch Buffer 2 Configuration</b> FPB is assigned to on chip bus master with master tag id equal to TAG2.
<b>TAG3</b>	21:16	rw	<b>Flash Prefetch Buffer 3 Configuration</b> FPB is assigned to on chip bus master with master tag id equal to TAG3.
<b>TAG4</b>	29:24	rw	<b>Flash Prefetch Buffer 4 Configuration</b> FPB is assigned to on chip bus master with master tag id equal to TAG4.

Table 130 Reset Values of FLASHCON0

Reset Type	Reset Value	Note
Application Reset	3F3F 3F3F <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

## CPUx Flash Configuration Register 1

The Flash Configuration Register 1 controls the stall response.

If a CPU instance does not have a local PFlash bank then the FLASHCON1 register associated with that instance will have no functionality.

## CPU Subsystem

### FLASHCON1

#### CPUX Flash Configuration Register 1

(01104<sub>H</sub>)

Application Reset Value: 0202 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES				RES1		RES				MASKUECC					
r					rw					r					rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES												STALL			
								r							rw

Field	Bits	Type	Description
STALL	0	rw	<b>Stall Bus Request</b> This field must not be changed while any bank is busy. The results are unpredictable. It is strongly recommended to configure this field once and avoid changing it during operation. Reading a flash bank, erase counter, or status register, in Sleep Mode must result in a bus error independent of this field (although it reports "busy"). 0 <sub>B</sub> <b>Error</b> , Reading local PFlash bank, erase counter, or status registers, when DMU_STATUS.PxBUSY is set, generates a bus error. 1 <sub>B</sub> <b>Stall</b> , Reading local PFlash bank, erase counter, or status registers when DMU_HF_STATUS.PxBUSY is set delays the response by inserting additional wait cycles until PxBUSY is cleared.
RES	15:1, 23:18, 31:26	r	<b>Reserved</b> Always read as 0; should be written with 0.
MASKUECC	17:16	rw	<b>Mask PFLASH Uncorrectable ECC Bit Error</b> No value other than 01 <sub>B</sub> or 10 <sub>B</sub> shall be programmed in the register bitfields. The system will behave as specified for 10 <sub>B</sub> if 00 <sub>B</sub> or 11 <sub>B</sub> is programmed. 01 <sub>B</sub> If a local PFLASH uncorrectable ECC error occurs, then the error is globally disabled for any requesting master reading the local PFLASH. 10 <sub>B</sub> If a PFLASH uncorrectable ECC error occurs then an uncorrectable ECC error is reported to the CPU.
RES1	25:24	rw	<b>Reserved - RES</b> Always read as 2; should be written with 2.

### CPUX Flash Configuration Register 2

The Flash Configuration Register 2 controls the recording of ECC errors and margin control.

No value other than 01<sub>B</sub> or 10<sub>B</sub> shall be programmed in the register bitfields.

If a CPU instance does not have a local PFlash bank then the FLASHCON2 register associated with that instance will have no functionality.

## CPU Subsystem

### FLASHCON2

#### CPUx Flash Configuration Register 2

(01108<sub>H</sub>)

Application Reset Value: AA02 0A0A<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ZBABCLR	MBABCLR	DBABCLR	SBABCLR	RES		ECCSCLR									
w	w	w	w	w		r								w	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		MSEL		HMARGIN		RES		ECCCORDIS		RECDIS					
r		rw		rw		r		rw		rw					

Field	Bits	Type	Description
RECDIS	1:0	rw	<b>Address Buffer Recording Disable</b> The system will behave as specified for 10 <sub>B</sub> if 00 <sub>B</sub> or 11 <sub>B</sub> is programmed, and an alarm will be generated. SBAB, DBAB, MBAB and ZBAB alarms will not be generated if RECDIS is 01 <sub>B</sub> . 01 <sub>B</sub> Disable local PFlash bank ECC error recording in SBAB, DBAB, MBAB and ZBAB. 10 <sub>B</sub> Enable local PFlash bank ECC error recording in SBAB, DBAB, MBAB and ZBAB.
ECCCORDIS	3:2	rw	<b>ECC Correction Disable</b> The system will behave as specified for 10 <sub>B</sub> if 00 <sub>B</sub> or 11 <sub>B</sub> is programmed, and an alarm will be generated on the next read. 01 <sub>B</sub> ECC correction for the local PFlash bank read path is disabled. 10 <sub>B</sub> ECC correction for the local PFlash bank read path is enabled.
RES	7:4, 15:12, 23:18	r	<b>Reserved</b> Always read as 0; should be written with 0.
HMARGIN	9:8	rw	<b>Hard Margin Selection</b> This register setting is effective only if FLASHCON2.MSEL = 01 <sub>B</sub> . The system will behave as specified for 10 <sub>B</sub> if 00 <sub>B</sub> or 11 <sub>B</sub> is programmed, and an alarm will be generated. 01 <sub>B</sub> Tight margin for 1 (high) level (sub-optimal 1 read as 0) for the local PFLASH. 10 <sub>B</sub> Tight margin for 0 (low) level (sub-optimal 0 read as 1) for the local PFLASH.
MSEL	11:10	rw	<b>Margin Read Selection</b> The system will behave as specified for 10 <sub>B</sub> if 00 <sub>B</sub> or 11 <sub>B</sub> is programmed, and an alarm will be generated. 01 <sub>B</sub> Read with the margin selected by FLASHCON2.HMARGIN for the local PFLASH. 10 <sub>B</sub> Read with the standard margin for the local PFLASH.

## CPU Subsystem

Field	Bits	Type	Description
<b>ECCSCLR</b>	17:16	w	<b>Clear ECC Status Register</b> The system will behave as specified for $10_B$ if $00_B$ or $11_B$ is programmed, and an alarm will be generated. $01_B$ Clear local PFlash bank ECC status register and the SBER and DBER alarms. $10_B$ No action.
<b>SBABCLR</b>	25:24	w	<b>Clear SBAB Record Registers</b> The system will behave as specified for $10_B$ if $00_B$ or $11_B$ is programmed, and an alarm will be generated. $01_B$ Clear local PFlash bank SBAB record registers and the SBAB alarm. $10_B$ No action.
<b>DBABCLR</b>	27:26	w	<b>Clear DBAB Record Registers</b> The system will behave as specified for $10_B$ if $00_B$ or $11_B$ is programmed, and an alarm will be generated. $01_B$ Clear local PFlash bank DBAB record registers and DBAB alarm. $10_B$ No action.
<b>MBABCLR</b>	29:28	w	<b>Clear MBAB Record Registers</b> The system will behave as specified for $10_B$ if $00_B$ or $11_B$ is programmed, and an alarm will be generated. $01_B$ Clear local PFlash bank MBAB record registers and the MBAB alarm. $10_B$ No action.
<b>ZBABCLR</b>	31:30	w	<b>Clear ZBAB Record Registers</b> The system will behave as specified for $10_B$ if $00_B$ or $11_B$ is programmed, and an alarm will be generated. $01_B$ Clear local PFlash bank ZBAB record registers and the ZBAB alarm. $10_B$ No action.

### CPUx Flash Configuration Register 3

The Flash Configuration Register 3 controls error injection.

If a CPU instance does not have a local PFlash bank then the FLASHCON3 register associated with that instance will have no functionality.

#### FLASHCON3

CPUx Flash Configuration Register 3 **(0110C<sub>H</sub>)** Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
RES																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
RES																
						<b>FLCON</b> <b>ERRIN</b> <b>J</b>	<b>NVMC</b> <b>ERRIN</b> <b>J</b>	<b>DBERE</b> <b>RRINJ</b>	<b>SBERE</b> <b>RRINJ</b>	<b>ZBABE</b> <b>RRINJ</b>	<b>MBAB</b> <b>ERRIN</b> <b>J</b>	<b>DBABE</b> <b>RRINJ</b>	<b>SBABE</b> <b>RRINJ</b>	<b>EDCER</b> <b>RINJ</b>	<b>ECCR</b> <b>RINJ</b>	
						r	rw	rw	rw	rw	rw	rw	rw	rw	rw	

## CPU Subsystem

Field	Bits	Type	Description
<b>ECCERRINJ</b>	0	rw	<b>ECC Error Injection</b> Setting this bit enforces an error in the ECC error correction supervision circuit. This can be used to check the correct function of this circuit. 0 <sub>B</sub> ECC logic operates normally. 1 <sub>B</sub> An error is injected into the ECC logic.
<b>EDCERRINJ</b>	1	rw	<b>EDC Error Injection</b> Setting this bit enforces an error in the EDC checker logic. This can be used to check the correct function of this circuit. 0 <sub>B</sub> EDC logic operates normally. 1 <sub>B</sub> An error is injected into the EDC checker logic.
<b>SBABERRINJ</b>	2	rw	<b>Corrected Single Bits Address Buffer (SBAB) Error Injection</b> Error injection logic to check the correct function of the SMU alarm. 0 <sub>B</sub> SBAB logic operates normally. 1 <sub>B</sub> Inject an SBAB alarm to the SMU.
<b>DBBABERRINJ</b>	3	rw	<b>Corrected Double Bits Address Buffer (DBAB) Error Injection</b> Error injection logic to check the correct function of the SMU alarm. 0 <sub>B</sub> DBAB logic operates normally. 1 <sub>B</sub> Inject an DBAB alarm to the SMU.
<b>MBBABERRINJ</b>	4	rw	<b>Uncorrected Multi Bit Address Buffer (MBAB) Error Injection</b> Error injection logic to check the correct function of the SMU alarm. 0 <sub>B</sub> MBAB logic operates normally. 1 <sub>B</sub> Inject an MBAB alarm to the SMU.
<b>ZBABERRINJ</b>	5	rw	<b>Uncorrected All Zeros Bits Address Buffer (ZBAB) Error Injection</b> Error injection logic to check the correct function of the SMU alarm. 0 <sub>B</sub> ZBAB logic operates normally. 1 <sub>B</sub> Inject an ZBAB alarm to the SMU.
<b>SBERERRINJ</b>	6	rw	<b>Single Bit Error (SBER) Injection</b> Setting this bit generates SBER SMU alarm, and can be used to check the correct function of this alarm. 0 <sub>B</sub> SBER logic operates normally. 1 <sub>B</sub> Inject a SBER alarm to the SMU.
<b>DBERERRINJ</b>	7	rw	<b>Double Bit Error (DBER) Injection</b> Setting this bit generates DBER SMU alarm and can be used to check the correct function of the DBER SMU alarm. 0 <sub>B</sub> DBER logic operates normally. 1 <sub>B</sub> Inject a DBER alarm to the SMU.
<b>NVMCERRINJ</b>	8	rw	<b>NVM Configuration (NVMCER) Injection</b> Setting this bit enforces an error from the error detection logic covering NVM configuration registers, and can be used to check the correct function of the NVMCER SMU alarm. 0 <sub>B</sub> NVM configuration logic operates normally. 1 <sub>B</sub> Inject a NVMCER alarm to the SMU.

## CPU Subsystem

Field	Bits	Type	Description
<b>FLCONERRINJ</b>	9	rw	<b>Flashcon Error (FLCONER) Injection</b> Setting this bit enforces an error from the error detection logic covering FLASHCON complement pair register bits in the PFRWB (RECDIS, ECCCORDIS, HMARGIN, MSEL and all CLR bits), and can be used to check the correct function of the FLCONER SMU alarm. 0 <sub>B</sub> FLASHCON register configuration operates normally. 1 <sub>B</sub> Inject a FLCONER alarm to the SMU.
<b>RES</b>	31:10	r	<b>Reserved</b> Always read as 0; should be written with 0.

### CPUx Flash Configuration Register 4

Direct access by the CPU to the Local Pflash Bank (LPB) may be disabled by setting the DDIS bit. When set all flash accesses are routed via the SRI bus system.

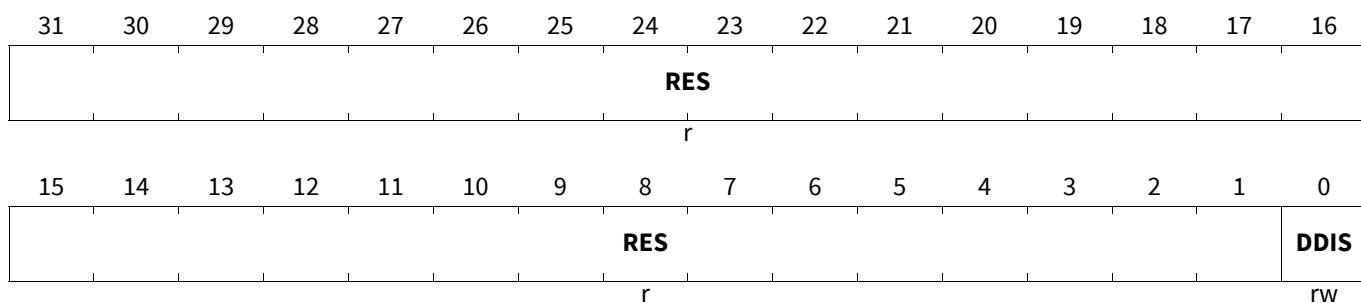
If a CPU instance does not have a local PFlash bank then the FLASHCON4 register associated with that instance will have no functionality.

#### FLASHCON4

##### CPUx Flash Configuration Register 4

(01110<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>DDIS</b>	0	rw	<b>Disable direct LPB access</b> Disable direct access by the CPU to the Local Pflash Bank (LPB). 0 <sub>B</sub> Direct access to the Local Pflash bank is enabled. 1 <sub>B</sub> Direct access to the Local Pflash bank is disabled.
<b>RES</b>	31:1	r	<b>Reserved</b> Always read as 0; should be written with 0.

### 5.3.8 Lockstep Comparator Logic (LCL)

The Lockstep Comparator Logic module provides access to the control and self test functions of the processor lockstep comparators.

#### 5.3.8.1 Feature List

An overview of the features implemented in the LCL follows:

- Monitoring the core comparators for the lockstep core and its shadow and flagging any detected differences
- Running background, continuous self test on the lockstep comparators to validate the correct operation of the logic.

## CPU Subsystem

### 5.3.8.2 Lockstep Control

The lockstep control function is enabled by the LSEN bitfield in a control register in the SCU. Each core capable of lockstep has its own instance of the control register.

These registers are only initialised by a cold power-on reset. In this initialisation state, all lockstepped processors in the system will have lockstep enabled. The lockstep function can only be disabled by the system initialisation software writing a  $0_B$  to the LSEN bitfield. Application software cannot enable or disable the lockstep function.

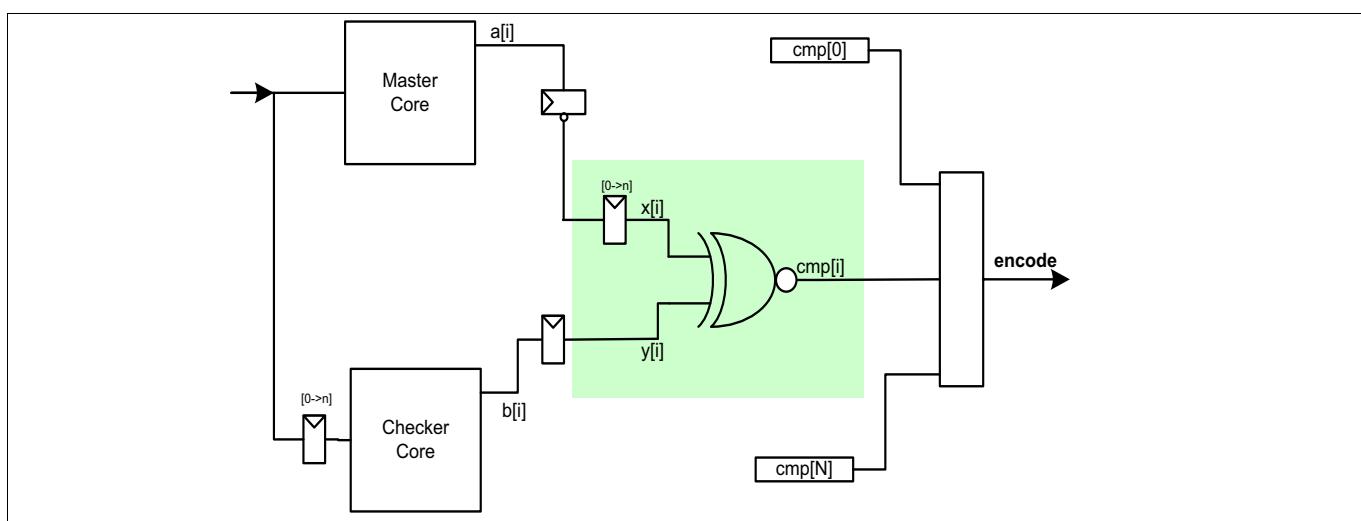
The current mode of the lockstep logic can be monitored by reading the lockstep status bit, LS, in the associated LCLCON register.

Writes to the control registers will be subject to the protection mechanisms of the SCU.

### 5.3.8.3 Lockstep Monitoring

The lockstep monitoring function will compare the outputs from the master and checker cores and report that a failure has occurred to the Safety Management Unit (SMU) for appropriate action.

The monitoring function temporally separates the cores by inserting synchronisation delays to re-align the signals being compared. To achieve this, the checker core inputs and the master core outputs fed to the comparators are delayed by two clock cycles.



**Figure 53 Node Comparator**

**Figure 53** above shows the equivalent circuit of an arbitrary node comparator,  $i$ , of 0 to  $N$  comparators. The comparator monitors two signals,  $\mathbf{a}$  and  $\mathbf{b}$ , which are connected to the same node in the master and checker cores. The signals can be synchronised in the relevant core using the core clock to minimise impact on the core timing.

After the optional synchronisation, the master core monitor point is inverted to reduce the risk of a common mode failure in the two monitored signals. Therefore,  $\mathbf{x}$  is equivalent to  $\mathbf{a}$  delayed by  $n$  clock cycles, where  $n=2$  in this implementation, to allow for the temporal shift between the master and checker cores and  $\mathbf{y}$  is equivalent to  $\mathbf{b}$ . If the nodes differ (i.e.  $\mathbf{x}$  and  $\mathbf{y}$  are the same), the CMP signal is set to  $1_B$ . This will cause the **encode** signal to flag the failing node and the failure will be detected. In the event that multiple nodes fail, the encode output will be the minimum and maximum values of all the indices,  $i$ , of the failing nodes. This has no impact on normal functioning but allows the self test logic to detect a real failure occurring in the same clock cycle that a fault is injected for test purposes.

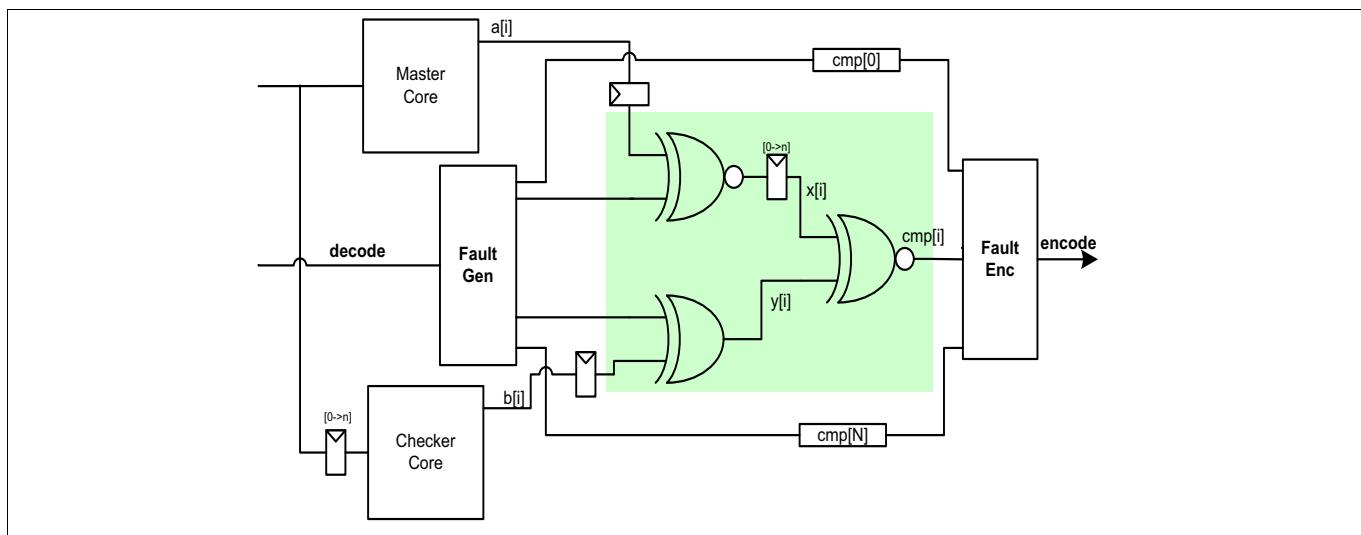
## CPU Subsystem

### 5.3.8.4 Lockstep Self Test

Each core capable of lockstep also has a continuously running background self test of the lockstep comparator. The self test function will inject faults into both inputs of each of the monitored nodes and verify that the fault is correctly detected by the monitoring logic.

In the event of a self test failure being detected, the failure will be reported to the Safety Management Unit for an appropriate response.

An equivalent circuit of a node comparator showing the fault injection logic is shown in [Figure 54](#)



**Figure 54 Node Comparator with Fault Injection**

Faults are injected using the **Fault Gen** block. This will use a binary number, **decode**, to calculate which node is to be tested. **decode** will be generated from a free running, binary counter using gray code number representation (the **input counter**). The Self test circuit will test every node once every 8192 clock cycles. Alternate test cycles will inject faults into either the **a** or **b** side of the comparator node. The complete self test cycle will therefore repeat every 16384 clock cycles.

Injecting a fault into either side of the comparator node will cause that node to fail unless a real fault occurs in the same clock cycle in which case the two faults will cancel out.

The node failing is processed by the **Fault Enc** block to generate a binary representation of the failing node, **encode**. This contains two numbers, the node index of the first node found to be failing counting up from the lowest minimum node index and the node index of the first node found to be failing counting down from the maximum node index.

If the lockstep block is functioning correctly, both values in **encode**, will either be 0 if no failures have been detected or the number of the node which has had a fault induced by the self test logic.

The values in **encode** are checked against a second, independent binary counter (the **monitor counter**). The **monitor counter** is also compared against the value of the **input counter**. In the event that either of the values in **encode** or the **input counter** fails to match the value of the monitor counter, a failure condition will be flagged to the SMU.

With this implementation, any of the following conditions will cause a self test fail:

- an actual fault occurring on any of the **a** or **b** nodes
- a stuck at  $0_B$  fault occurring on any of the **cmp** nodes
- a stuck at  $1_B$  fault occurring on any of the **cmp** nodes
- a failure in the **Fault Gen** block causing an incorrect or no fault to be injected

## CPU Subsystem

- a failure in the **Fault Enc** block causing an incorrect detection or no fault to be detected
- a stuck at fault on **x** (assuming that an injected fault does not always coincide with a masking pulse on **b**)
- a stuck at fault on **y** (assuming that an injected does not always coincide with a masking pulse on **a**)
- an actual fault on any of the **a** or **b** nodes coinciding with an injected fault
- a soft error occurring on either the **input counter** or **monitor counter**.

### 5.3.8.5 Lockstep Failure Signalling Test

The lockstep comparator allows a failure to be injected into one of the comparator nodes to allow the signalling of failures to be verified. A failure can be injected by writing  $1_B$  to the LCLT bitfield of the LCLTEST register. The failure will be injected for a single cycle of the SPB clock. It is not necessary to write a  $0_B$  to clear the test.

### 5.3.8.6 Functional Redundancy

All registers in the lockstep block which are not capable of being directly monitored for correct operation by the self test function will be duplicated. In the event of the duplicated registers not storing the same state, an error will be flagged to the SMU.

---

**CPU Subsystem**

### 5.3.9 Data Access Overlay (OVC)

The data overlay provides the capability to redirect selected data accesses to the Overlay memory. Data accesses made by the TriCore to Program Flash, Online Data Acquisition space, or EBU space (if present) can be redirected. Overlay memory may be located in the Local Memory (if present), in the Emulation Memory (Emulation Device only), in the EBU space, or in the DPSR/PSPR memory. Overlay functionality makes it possible, for example, to modify the application's test and calibration parameters (which are typically stored in Flash memory) during run time of a program. Note that only read and write data accesses are redirected. The access redirection is executed without performance penalty.

**Attention:** As the address translation is implemented in the DMI it is only effective for data accesses by the TriCore. Instruction fetches by the TriCore or accesses by any other master (including the debug interface) are not effected!

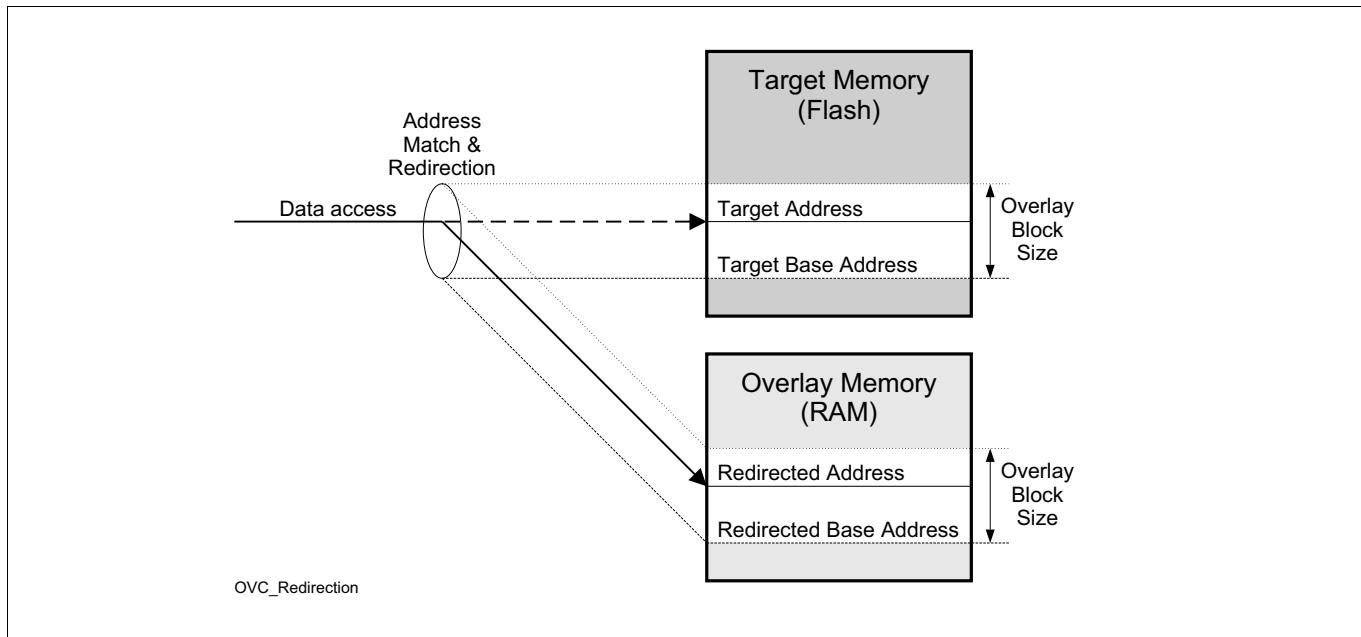
#### Summary of Features and Functions

- Redirecting data accesses addressed to Program Flash, OLDA or External EBU Space.
- Support redirection to Overlay memory located in:
  - Local Memory (LMU) (if present)
  - Emulation Memory (Emulation Device only)
  - EBU space (if present)
  - DPSR or PSPR memory
- Support of up to 4 MB overlay memory address range;
- Up to 32 overlay ranges ("blocks") available in each TriCore instance;
- Overlay block size from 32 byte to 128 Kbyte;
- Overlay memory location and block size selected individually for every overlay block;
- Multiple overlay blocks can be enabled or disabled with one register write access;
- Programmable flush (invalidate) control for data cache in DMI.
- Overlay start/stop synchronised against data load.
- Individual Overlay system per processor core.

## CPU Subsystem

### 5.3.9.1 Data Access Redirection

The principle of redirecting data access from the original target memory (“Target Address”) to overlay memory (“Redirected Address”) is shown below.



**Figure 55 Data Access Redirection**

Data access overlay is defined using overlay ranges (“overlay blocks”). Each overlay block defines one continuous range of address space for which the accesses are redirected. Each overlay block is configured with the following parameters:

- Overlay Block Target Base Address - the start address for the range of the target addresses to be redirected;
- Overlay Block Size - the size of the range of the addresses to be redirected;
- Overlay Block Redirection Base Address - the start address for redirection.

In AURIX™ up to 32 overlay ranges can be used in each TriCore instance.

Each overlay block has 3 associated registers for independent configuration of these parameters. The overlay parameters are configured as follows:

- Target Base Address is configured with OTARx register,
- Overlay Block Size is configured with OMASKx register,
- Redirection Base Address is configured with RABRx register.

The size of the overlay memory blocks can be  $2^n \times 32$  bytes, with  $n = 0$  to 12. This gives the block size range from 32 bytes to 128 Kbytes. The start address of the block can only be an integer multiple of the programmed block size (natural alignment boundary). If OTAR register value, or RABR register value is not aligned with the block size, the least significant bit values are ignored and treated as zero.

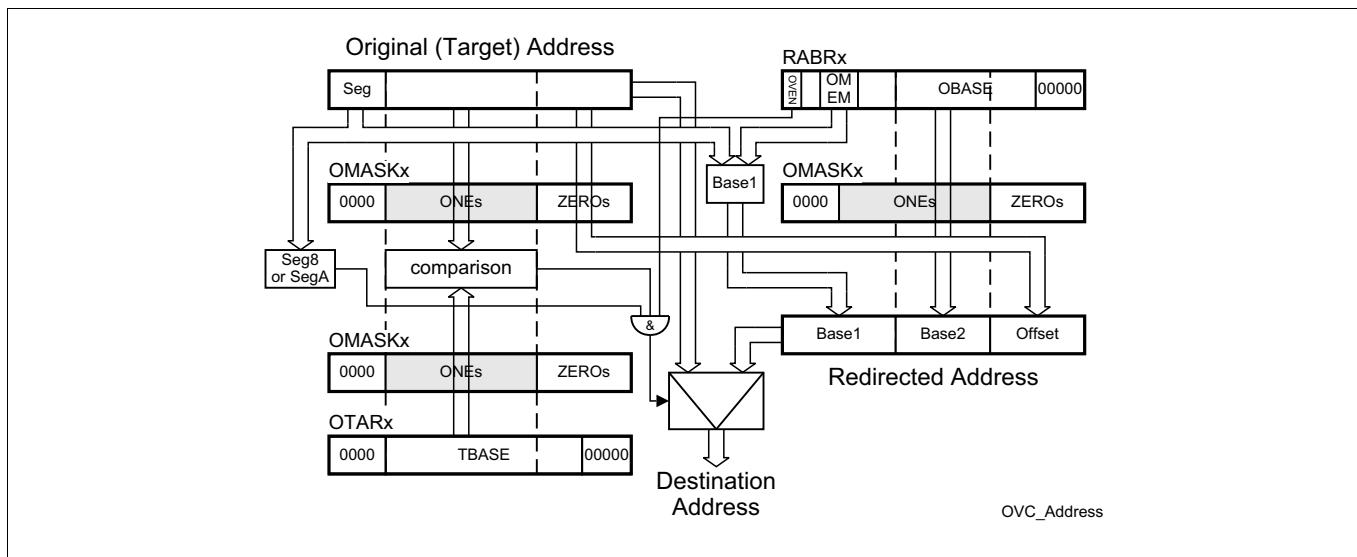
The Redirection Base Address is determined by two fields in RABRx register:

- RABRx.OMEM selecting the Overlay Memory, and
- RABRx.OBASE selecting the base address within this memory.

Each overlay block can be activated or deactivated with its RABRx.OVEN bit. Overlay blocks can be activated and deactivated independently, by directly accessing RABRx register, or in groups, where multiple configured blocks are activated or deactivated concurrently. For information on concurrent block activation see [Chapter 5.3.9.4.1](#).

The address redirection process is shown in the following figure.

## CPU Subsystem



**Figure 56 Address Redirection Process**

Any data access to segment  $8_H$  or segment  $A_H$  is checked against all the activated overlay blocks. For each activated overlay block, address bits 27..5 are compared with the target base address (OTARx), and this bit-wise comparison is qualified by the content of OMASKx register. Address bits participate in the comparison if the corresponding OMASKx bits are set to one. The access is redirected, if all the address bits selected by OMASKx equal to the corresponding bits in OTARx register.

The address for redirection is constructed as follows:

- Address bits 31..22 are set according to the overlay memory selection (RABRx.OMEM) and the cache-ability of the original address.
- For address bits 21..5:
  - If the corresponding OMASKx bit is set, the address bit value is taken from RABRx.OBASE field;
  - If the corresponding OMASKx bit is cleared, the address bit value is taken from the original address.
- Address bits 4..0 are always taken directly from the original address.

If there is no redirection, the original address is used to perform the access.

Target address ranges for activated overlay blocks should not overlap or an exception may occur, see [Chapter 5.3.9.6](#).

### 5.3.9.2 Target Memories

Data access to any memory within the segment  $8_H$  or segment  $A_H$  may be redirected to overlay memory. In particular, access to the following memories may be redirected:

- Program Flash;
- Data Flash;
- OLDA space;
- External EBU space.

#### 5.3.9.2.1 Online Data Acquisition (OLDA) Space

Calibration is additionally supported by virtual OLDA memory range. The base address of the virtual OLDA memory range is  $A/8FE7\ 0000_H$ .

## CPU Subsystem

### 5.3.9.3 Overlay Memories

In the following, all the supported overlay memories are described. Note, that depending on the device type only a subset of the listed overlay memories is available. Overlay memory is selected independently for each block by RABRx.OMEM field value.

#### 5.3.9.3.1 Local Memory

If present, the Local Memory (LMU) can be selected for overlay. The Local Memory is selected for overlay block x redirection, if RABRx.OMEM value is 8. The base address of the LMU is  $B/9000\ 0000_H$ . During address translation, the upper 10 address bits are set to  $B0_H00_B$  (for target segment  $A_H$ ) or to  $90_H00_B$  (for target segment  $8_H$ ).

#### 5.3.9.3.2 External Memory

If present, the External Memory (EBU space) can be selected for overlay. The EBU space is selected for overlay block x redirection, if RABRx.OMEM value is  $A_H$ . The base address of the EBU space is  $A/8100\ 0000_H$ . During address translation, the upper 10 address bits are set to  $A1_H00_B$  (for target segment  $A_H$ ) or to  $81_H00_B$  (for target segment  $8_H$ ).

#### 5.3.9.3.3 DSPR & PSPR Memory

Data Scratch Memory (DSPR) or Program Scratch Memory (PSPR) from any core can be selected for overlay by configuring the block RABRx.OMEM value. During address translation, the upper 10 address bits are set to the target PSPR or DSPR target address. Depending on the value set in RABRx.OBASE either DSPR memory (starting at offset  $0_H$ ) or PSPR memory (starting at offset  $10\ 0000_H$ ) can be used.

### 5.3.9.4 Global Overlay Control

Overlay can be disabled or enabled individually for each core with OVCENABLE register. If OVCENABLE.OVENn bit is cleared no address redirection is permitted on Core-n regardless of the remaining register settings. A write to OVCENABLE register does not change any of the remaining register values.

While each overlay block can be activated and deactivated individually by writing its RABRx.OVEN bit, a dedicated functionality is provided for concurrently activating and deactivating multiple blocks. This can be useful in maintaining data consistency across several memory regions. For the purpose of concurrent activation and deactivation overlay blocks are selected in two stages:

- The individual blocks for activation and deactivation are selected with OVCn\_OSEL registers, for each core-n independently;
- The set of cores is selected with OVCCON.CSEL field.

Multiple overlay blocks can be simultaneously activated or deactivated with OVCCON.OVSTRT bit. When OVCCON.OVSTRT bit is written with one:

- If OVCCON.CSELn bit is written with one, and OVCn\_OSEL.SHOVENx bit value is one, overlay block x in core-n is activated, and OVCn\_RABRx.OVEN bit is set;
- If OVCCON.CSELn bit is written with one, and OVCn\_OSEL.SHOVENx bit value is zero, overlay block x in core-n is deactivated, and OVCn\_RABRx.OVEN bit is cleared;
- If OVCCON.CSELn bit is written with zero, the overlay configuration in core-n is not effected.

The actions listed above are executed concurrently. The overlay configuration is not changed otherwise. With this function it is possible to switch directly from one set of overlay blocks to another set of overlay blocks.

Multiple overlay blocks can be simultaneously deactivated with OVCCON.OVSTP bit. When OVCCON.OVSTP is written with one:

## CPU Subsystem

- If OVCCON.CSELn bit is written with one, all the overlay blocks in core-n are deactivated, and all OVCn\_RABRx.OVEN bits are cleared;
- If OVCCON.CSELn bit is written with zero, the overlay configuration in core-n is not effected.

The actions listed above are executed concurrently. The overlay configuration is not changed otherwise.

**Note:** *Overlay should not be enabled or disabled using global OVCENABLE register if any of the blocks are enabled with RABRx.OVEN. Instead, OVSTRT or OVSTP should be used if concurrent block enabling or disabling is required.*

When OVCCON.DCINVAL is written with one, all the unmodified (clean) data cache lines in the selected cores are invalidated. The data cache lines containing modified (dirty) data are not effected. The cores not selected with OVCCON.CSEL field are not effected. Data Cache invalidation can be combined with OVSTRT or OVSTP action. This function helps to assure that the CPU can access the new data after the overlay blocks have been activated or deactivated.

**Note:** *OVCCON.CSEL field is written together with OVSTRT, OVSTP and DCINVAL bits in the same register, and only impacts any action triggered by the same write. CSEL, OVSTRT, OVSTP and DCINVAL do not retain the written value and always read as zero.*

The OVCCON.OVCONF user control flag is provided, together with its protection bit OVCCON.POVOCONF. These bits do not impact the overlay functionality.

When OVCCON register is written with CSELn set and either OVSTRT or OVSTP set, and at the same time OVCn\_RABRy register is written, the resulting value of OVCn\_RABRx.OVEN bit is not defined. Possibility of such simultaneous access should be avoided.

OVSTRT, OVSTP and DCINVAL actions are not performed when TriCore is in IDLE state.

### 5.3.9.4.1 Global Overlay Control Synchronisation

When OVSTRT, OVSTP or DCINVAL action is requested its execution may be delayed to prevent changing Overlay configuration during an ongoing data load.

Sufficient time should be allowed after an action request, before another action is requested. If a new action is requested while previous action is still pending (due to synchronisation with CPU loads) some actions may be lost.

### 5.3.9.5 Overlay Configuration Change

Overlay block should be disabled, by clearing its RABRx.OVEN bit, before any changes are made to OTARx, OMASKx or RABRy registers. Otherwise, unintended access redirections may occur. Overlay block should only be enabled, if the target address, the overlay memory selection, the redirection address and the mask have all been configured with intended values.

**Note:** *The Overlay Control does not prevent configuring the translation logic incorrectly. In particular, redirection to not implemented or forbidden address range is not prevented.*

Special care needs to be taken to synchronise Overlay redirection change to the executed instruction stream if data consistency is required.

External accesses may be buffered in the CPU. External accesses that have not been completed may still be affected by overlay configuration changes. Therefore, it is advised to ensure completion of all pending accesses (for example, by executing DSYNC instruction) before any overlay range is activated or deactivated.

---

## CPU Subsystem

When overlay block is enabled and the same memory location is written through the target address space and read through the redirected address space, or vice-versa, the access synchronisation need to be enforced (with DSYNC and data cache writeback if applicable).

### 5.3.9.6 Access Protection, Attributes, Concurrent Matches

When data access is redirected by Overlay, access protection is applied as follows:

- Target address is subject to CPU Memory Protection (MPU) validation;
- Redirected address is subject to Safety Protection validation.

Physical Memory Attributes for the redirected access are determined basing on the Target Address segment (see CPU Physical Memory Attributes chapter).

Concurrent matches in more than one enabled overlay block are not supported. When an address matches two, or more, of the enabled overlay blocks, an exception is raised and the memory access is not performed. A load operation with multiple matches on overlay ranges, raises a Data Access Synchronous Error (DSE) trap, and a store operation raises Data Access Asynchronous Error (DAE) trap. In such case, relevant trap information registers: Data Synchronous Trap Register (DSTR), Data Asynchronous Trap Register (DATR), and Data Error Address Register (DEADD) are updated, see DMI Registers chapter for more information.

## CPU Subsystem

### 5.3.9.7 Overlay Control Registers

OVC block control registers are located in each module that supports data access overlay. OVC global control registers are located in SCU. OVC register access can be restricted by Safety Register Protection.

#### Per Core Overlay register

- OVCn\_RABRx
- OVCn\_OTARx
- OVCn\_OMASKx
- OVCn\_OSEL

Where n is the core number and x is the block number.

#### Global Overlay Registers

- OVCENABLE
- OVCCON

Registers OVCENABLE and OVCCON are described in SCU Chapter.

**Table 131 Register Overview - OVERLAY\_BLOCK\_CONTROL (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
OSEL	CPUx Overlay Range Select Register	0FB00 <sub>H</sub>	U,SV,32	SV,32,P	Application Reset	<a href="#">111</a>
RABRI	CPUx Redirected Address Base Register i	0FB10 <sub>H</sub> +i *12	U,SV,32	SV,32,P	Application Reset	<a href="#">111</a>
OTARI	CPUx Overlay Target Address Register i	0FB14 <sub>H</sub> +i *12	U,SV,32	SV,32,P	Application Reset	<a href="#">112</a>
OMASKi	CPUx Overlay Mask Register i	0FB18 <sub>H</sub> +i *12	U,SV,32	SV,32,P	Application Reset	<a href="#">113</a>

#### 5.3.9.7.1 Block control registers

For each of the 32 overlay memory blocks (indicated by index x), three registers control the overlay operation and the memory selection:

- Redirected Address Base Register RABRx, which selects the overlay memory, holds the block base address within this memory, and contains block enable bit.
- Overlay Target Address Register OTARx, which holds the base address of the memory block being overlaid.
- Overlay Mask Register OMASKx, which determines which bits (from RABRx) are used for the base address (of overlay memory and block) and which bits (of original data address) are directly used as offset within the block.

Additionally, Overlay Range Select Registers OSEL determines which blocks are to be enabled and which blocks are to be disabled when OVCCON.OVSTRT bit is set.

All overlay block control registers are reset to their default values with the application reset. A special debug reset is not considered.

**CPU Subsystem****CPUX Overlay Range Select Register****OSEL**
**CPUX Overlay Range Select Register** **(0FB00<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>SHOVEN_x</b>															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>SHOVEN_x</b>															
rw															

Field	Bits	Type	Description
<b>SHOVEN_x</b>	31:0	rw	<b>Shadow Overlay Enable - SHOVEN[x]</b> One enable bit is provided for each of the 32 overlay blocks. 00000000 <sub>H</sub> Overlay block x is disabled when OVCCON.OVSTRT is set. 00000001 <sub>H</sub> Overlay block x is enabled when OVCCON.OVSTRT is set.

**CPUX Redirected Address Base Register i****RABR<sub>i</sub> (i=0-31)**
**CPUX Redirected Address Base Register i** **(0FB10<sub>H</sub>+i\*12)** **Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>OVEN</b>	<b>RES</b>			<b>OMEM</b>			<b>RES</b>			<b>OBASE</b>					
rwh	r			rw			r			rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>OBASE</b>														<b>RES</b>	
rw															

Field	Bits	Type	Description
<b>RES</b>	4:0, 23:22, 30:28	r	<b>Reserved</b> Reads as 0; should be written with 0.
<b>OBASE</b>	21:5	rw	<b>Overlay Block Base Address</b> Bits 21..5 of the base address the overlay memory block in the overlay memory. If the corresponding bit in OMASK register is set to one, OBASE bit value is used in the redirection address. If the corresponding bit in OMASK register is set to zero, OBASE bit value is ignored.

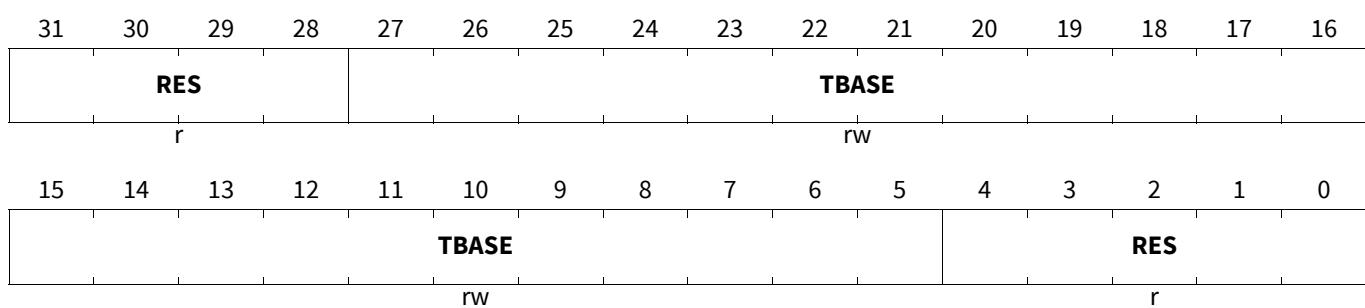
## CPU Subsystem

Field	Bits	Type	Description
<b>OMEM</b>	27:24	rw	<b>Overlay Memory Select</b> Selects overlay memory used for redirection. 0 <sub>H</sub> Redirection to Core 0 DSPR/PSPR memory 1 <sub>H</sub> Redirection to Core 1 DSPR/PSPR memory 2 <sub>H</sub> Redirection to Core 2 DSPR/PSPR memory 3 <sub>H</sub> Redirection to Core 3 DSPR/PSPR memory 4 <sub>H</sub> Redirection to Core 4 DSPR/PSPR memory 5 <sub>H</sub> Redirection to Core 5 DSPR/PSPR memory 6 <sub>H</sub> Reserved 7 <sub>H</sub> Reserved 8 <sub>H</sub> Redirection to LMU 9 <sub>H</sub> Redirection to EMEM A <sub>H</sub> Redirection to EBU B <sub>H</sub> Reserved ... F <sub>H</sub> Reserved
<b>OVEN</b>	31	rwh	<b>Overlay Enabled</b> This bit controls whether the overlay function of overlay block x is enabled. This bit can also be changed when OVCCON.OVSTP or OVCCON.OVSTRT is set. See OVCCON register description. 0 <sub>B</sub> Overlay function of block x is disabled. 1 <sub>B</sub> Overlay function of block x is enabled.

### CPUx Overlay Target Address Register i

OTARI (i=0-31)

CPUx Overlay Target Address Register i

(0FB14<sub>H</sub>+i\*12)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>RES</b>	4:0, 31:28	r	<b>Reserved</b> Reads as 0; should be written with 0.

## CPU Subsystem

Field	Bits	Type	Description
TBASE	27:5	rw	<p><b>Target Base</b></p> <p>This field holds the base address of the overlay memory block in the target memory.</p> <p>If the corresponding bit in OMASK register is set to one TBASE bit value is used in the address match.</p> <p>If the corresponding bit in OMASK register is set to zero TBASE bit value is ignored.</p>

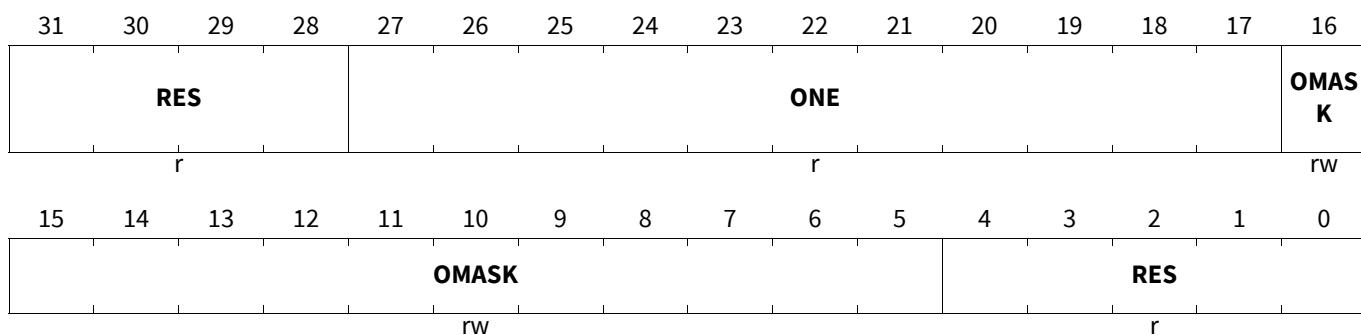
### CPUx Overlay Mask Register i

**OMASKi (i=0-31)**

**CPUx Overlay Mask Register i**

**(0FB18<sub>H</sub>+i\*12)**

**Application Reset Value: 0FFF FFE0<sub>H</sub>**



Field	Bits	Type	Description
<b>RES</b>	4:0, 31:28	r	<p><b>Reserved</b></p> <p>Corresponding address bits are not used in the address comparison.</p> <p>Corresponding final address bits are taken from the original data address.</p>
<b>OMASK</b>	16:5	rw	<p><b>Overlay Address Mask</b></p> <p>This bitfield determines the overlay block size and the bits used for address comparison and translation.</p> <p>[...]</p> <p>“Zero” bits determine the corresponding address bits which are not used in the address comparison and thus determine the block size; corresponding final address bits are derived from the original data address.</p> <p>“One” bits determine the corresponding address bits which are used for the address comparison; corresponding final address bits are derived from RABRx register in case of address match.</p> <p>000<sub>H</sub> , 128 Kbyte block size            800<sub>H</sub> , 64 Kbyte block size            C00<sub>H</sub> , 32 Kbyte block size            FFE<sub>H</sub> , 64 byte block size            FFF<sub>H</sub> , 32 byte block size</p>
<b>ONE</b>	27:17	r	<p><b>Fixed “1” Values</b></p> <p>Corresponding address bits are participating in the address comparison.</p> <p>Corresponding final address bits are taken from RABRx.</p>

---

**CPU Subsystem**

### 5.3.9.8 Global overlay control registers

Two registers globally control the overlay operation for all the cores:

- Overlay Enable Register OVCENABLE can be used to disable or enable data access overlay individually for each core;
- Overlay Control Register OVCCON can be used to perform the following action on selected set of cores:
  - concurrently enable / disable selected overlay blocks,
  - concurrently disable overlay blocks,
  - invalidate data cache.

---

## CPU Subsystem

### 5.3.10 CPU Architecture registers

#### 5.3.10.1 Registers with architecturally defined reset values

Please refer to the Tricore Architecture Manual for the descriptions and reset values of the following registers.

- SYSCON
- CPU\_SYSCON
- ICR
- CPU\_ICR
- TRiEVT (i=0-7)
- CPU\_TRiEVT (i=0-7)
- TRiADR (i=0-7)
- CPU\_TRiADR (i=0-7)
- CCTRL
- CPU\_CCTRL
- CCNT
- CPU\_CCNT
- ICNT
- CPU\_ICNT
- M1CNT
- CPU\_M1CNT
- M2CNT
- CPU\_M2CNT
- M3CNT
- CPU\_M3CNT
- DBGSR
- CPU\_DBGSR
- EXEVT
- CPU\_EXEVT
- CREVT
- CPU\_CREVT
- SWEVT
- CPU\_SWEVT
- TRIG\_ACC
- CPU\_TRIG\_ACC
- DBGTMR
- CPU\_DBGTMR

#### 5.3.10.2 Program Counter (PC)

Please refer to the Tricore Architecture Manual for the description of the Program Counter (PC) register. The reset value of the PC is configured by the SSW. For details on the reset value please refer to the Firmware chapter (Boot Mode evaluation sequence)

## CPU Subsystem

- PC
- CPU\_PC

### 5.3.10.3 Registers with Implementation specific reset values

The reset values and description of CPU architecture registers with implementation specific reset values can be listed in this section.

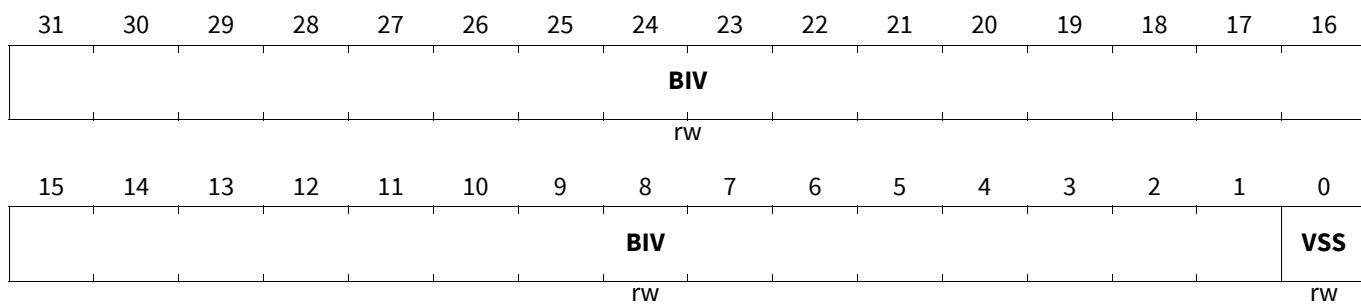
#### CPUX Base Interrupt Vector Table Pointer

**BIV**

**CPUX Base Interrupt Vector Table Pointer** (1FE20<sub>H</sub>) **Application Reset Value:** 0000 0000<sub>H</sub>

**CPU\_BIV**

**Short address for domain CSFR** (0FE20<sub>H</sub>) **Application Reset Value:** 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>VSS</b>	0	rw	<b>Vector Spacing Select</b> 0: 32 byte vector spacing. 1: 8 Byte vector spacing.
<b>BIV</b>	31:1	rw	<b>Base Address of Interrupt Vector Table</b> The address in the BIV register must be aligned to an even byte address (halfword address). Because of the simple ORing of the left-shifted priority number and the contents of the BIV register, the alignment of the base address of the vector table must be to a power of two boundary, dependent on the number of interrupt entries used. For the full range of 256 interrupt entries an alignment to an 8 KByte boundary is required. If fewer sources are used, the alignment requirements are correspondingly relaxed.

## CPU Subsystem

### CPUx Base Trap Vector Table Pointer

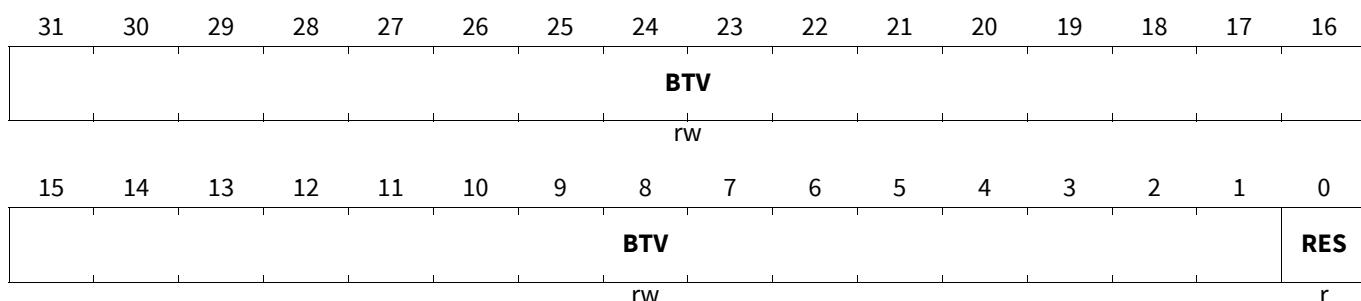
BTV

CPUx Base Trap Vector Table Pointer

(1FE24<sub>H</sub>)Application Reset Value: A000 0100<sub>H</sub>

CPU\_BTV

Short address for domain CSFR

(0FE24<sub>H</sub>)Application Reset Value: A000 0100<sub>H</sub>

Field	Bits	Type	Description
RES	0	r	<b>Reserved</b> Read as 0; should be written as 0.
BTW	31:1	rw	<b>Base Address of Trap Vector Table</b> The address in the BTV register must be aligned to an even byte address (halfword address). Also, due to the simple ORing of the left-shifted trap identification number and the contents of the BTV register, the alignment of the base address of the vector table must be to a power of two boundary. There are eight different trap classes, resulting in Trap Classes from 0 to 7. The contents of BTV should therefore be set to at least a 256 byte boundary (8 Trap Classes * 8 word spacing).

### CPUx Interrupt Stack Pointer

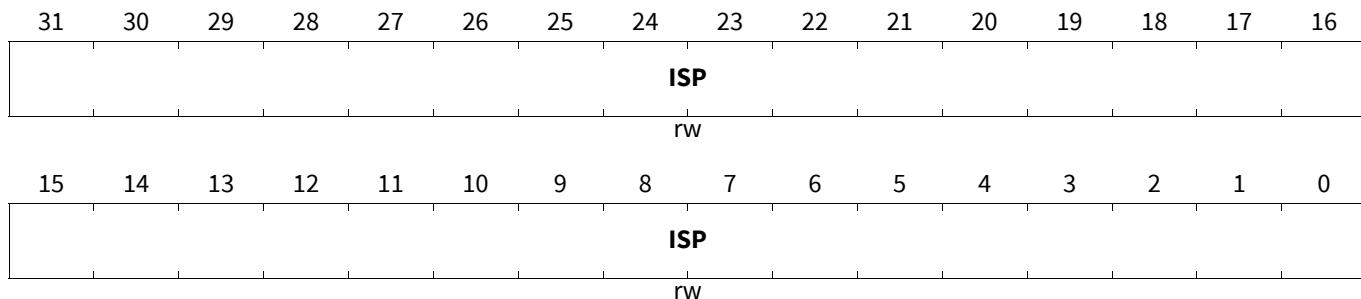
ISP

CPUx Interrupt Stack Pointer

(1FE28<sub>H</sub>)Application Reset Value: 0000 0100<sub>H</sub>

CPU\_ISP

Short address for domain CSFR

(0FE28<sub>H</sub>)Application Reset Value: 0000 0100<sub>H</sub>

Field	Bits	Type	Description
ISP	31:0	rw	<b>Interrupt Stack Pointer</b>

## CPU Subsystem

### CPUx Free CSA List Head Pointer

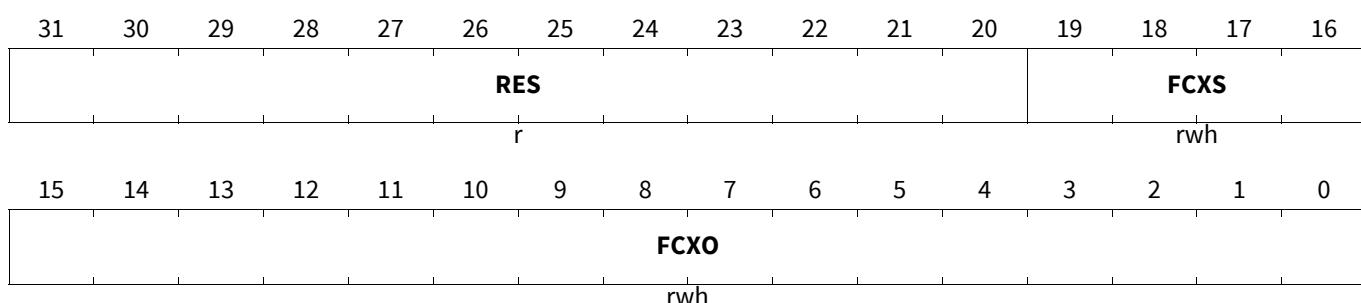
FCX

CPUx Free CSA List Head Pointer

**(1FE38<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

CPU\_FCX

Short address for domain CSFR

**(0FE38<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
FCXO	15:0	rwh	<b>FCX Offset Address Field</b> The FCXO and FCXS fields together form the FCX pointer, which points to the next available CSA.
FCXS	19:16	rwh	<b>FCX Segment Address Field</b> Used in conjunction with the FCXO field.
RES	31:20	r	<b>Reserved</b> Read as 0; should be written as 0.

### CPUx Free CSA List Limit Pointer

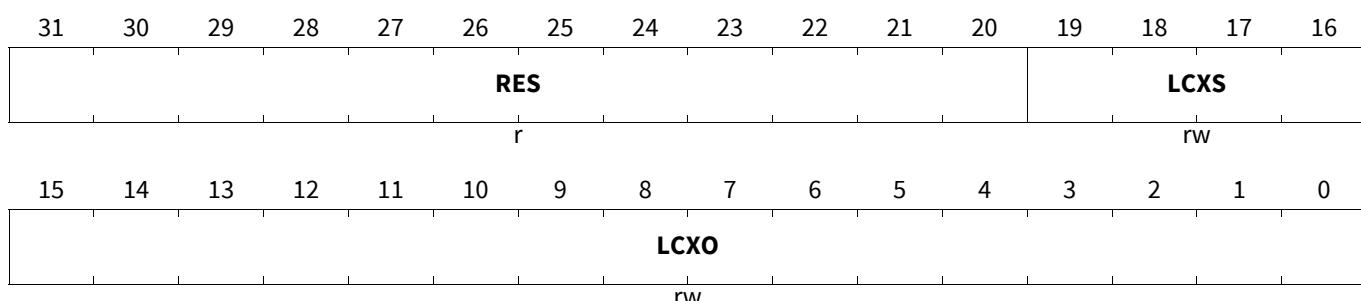
LCX

CPUx Free CSA List Limit Pointer

**(1FE3C<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

CPU\_LCX

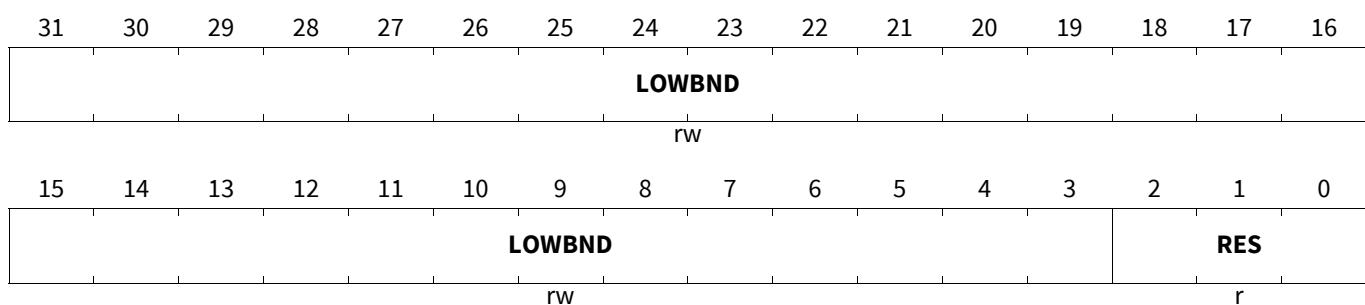
Short address for domain CSFR

**(0FE3C<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

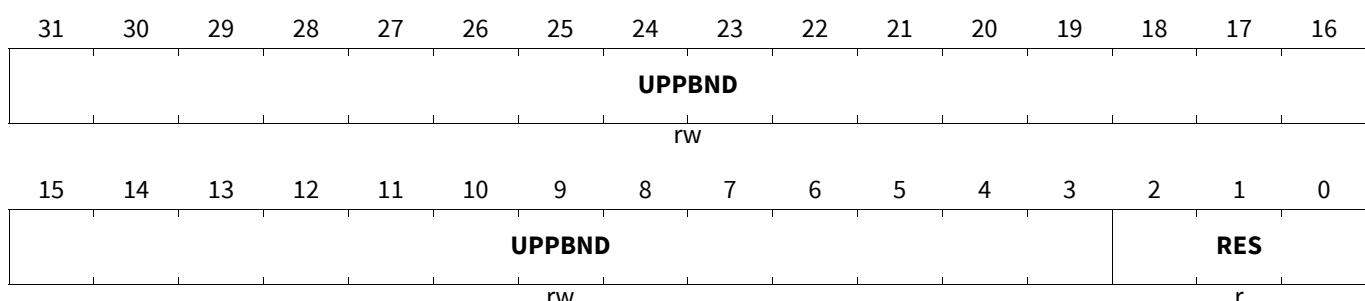
Field	Bits	Type	Description
LCXO	15:0	rw	<b>LCX Offset Field</b> The LCXO and LCXS fields form the pointer LCX, which points to the last available CSA.
LCXS	19:16	rw	<b>LCX Segment Address</b> This field is used in conjunction with the LCXO field.

**CPU Subsystem**

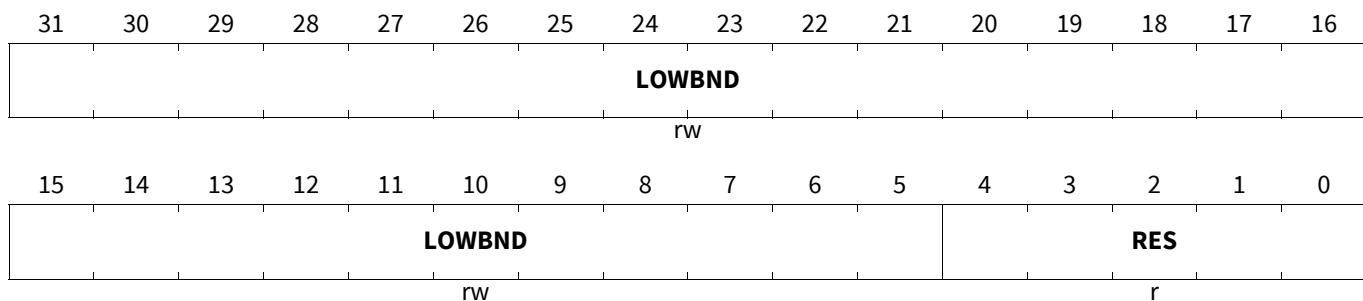
Field	Bits	Type	Description
<b>RES</b>	31:20	r	<b>Reserved</b> Read as 0; should be written as 0.

**CPUx Data Protection Range y, Lower Bound Register****DPRy\_L (y=0-17)****CPUx Data Protection Range y, Lower Bound Register( $1C000_H+y*8$ ) Application Reset Value: 0000 0000<sub>H</sub>****CPU\_DPRy\_L (y=0-17)****Short address for domain CSFR (0C000<sub>H</sub>+y\*8) Application Reset Value: 0000 0000<sub>H</sub>**

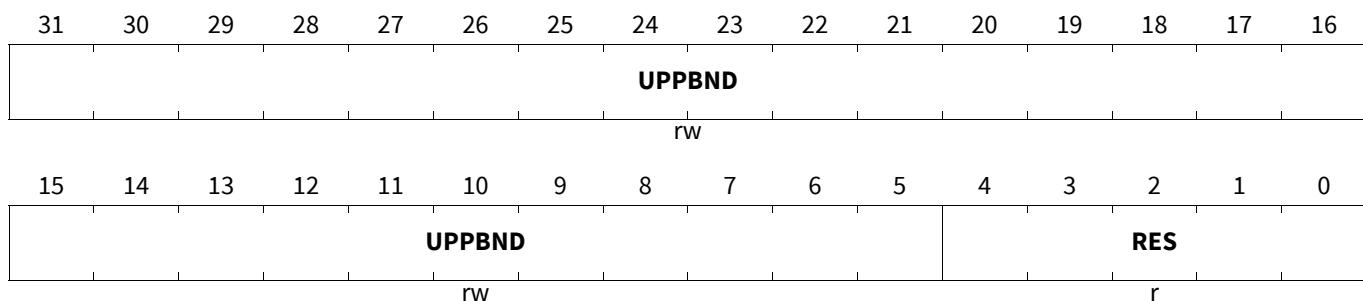
Field	Bits	Type	Description
<b>RES</b>	2:0	r	<b>Reserved</b> The three least significant bits are not writable and always return zero.
<b>LOWBND</b>	31:3	rw	<b>DPRy Lower Boundary Address</b>

**CPUx Data Protection Range y, Upper Bound Register****DPRy\_U (y=0-17)****CPUx Data Protection Range y, Upper Bound Register( $1C004_H+y*8$ ) Application Reset Value: 0000 0000<sub>H</sub>****CPU\_DPRy\_U (y=0-17)****Short address for domain CSFR (0C004<sub>H</sub>+y\*8) Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>RES</b>	2:0	r	<b>Reserved</b> The three least significant bits are not writable and always return zero.
<b>UPPBND</b>	31:3	rw	<b>DPRy Upper Boundary Address</b>

**CPU Subsystem****CPUx Code Protection Range y Lower Bound Register****CPRy\_L (y=0-9)****CPUx Code Protection Range y Lower Bound Register( $1D000_H+y*8$ ) Application Reset Value: 0000 0000<sub>H</sub>****CPU\_CPRy\_L (y=0-9)****Short address for domain CSFR****(0D000<sub>H</sub>+y\*8)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>RES</b>	4:0	r	<b>Reserved</b> The 5 least significant bits are not writable and always return zero.
<b>LOWBND</b>	31:5	rw	<b>CPRy Lower Boundary Address</b>

**CPUx Code Protection Range y Upper Bound Register****CPRy\_U (y=0-9)****CPUx Code Protection Range y Upper Bound Register( $1D004_H+y*8$ ) Application Reset Value: 0000 0000<sub>H</sub>****CPU\_CPRy\_U (y=0-9)****Short address for domain CSFR****(0D004<sub>H</sub>+y\*8)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>RES</b>	4:0	r	<b>Reserved</b> The 5 least significant bits are not writable and always return zero.
<b>UPPBND</b>	31:5	rw	<b>CPR0_m Upper Boundary Address</b>

## CPU Subsystem

### CPUx Code Protection Execute Enable Register Set y

**CPXE\_y (y=0-3)**

**CPUx Code Protection Execute Enable Register Set y( $1E000_H+y*4$ ) Application Reset Value: 0000 0000<sub>H</sub>**

**CPXE\_y (y=4-5)**

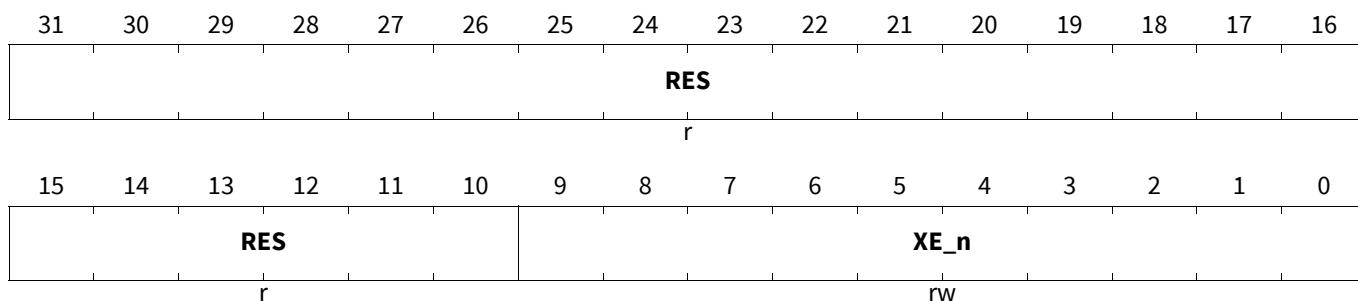
**CPUx Code Protection Execute Enable Register Set y( $1E040_H+(y-4)*4$ ) Application Reset Value: 0000 0000<sub>H</sub>**

**CPU\_CPXЕ\_y (y=0-3)**

**Short address for domain CSFR ( $0E000_H+y*4$ ) Application Reset Value: 0000 0000<sub>H</sub>**

**CPU\_CPXЕ\_y (y=4-5)**

**Short address for domain CSFR ( $0E040_H+(y-4)*4$ ) Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>XE_n</b>	9:0	rw	<b>Execute Enable Range select - XE[n]</b> 000 <sub>H</sub> Code Protection Range-n not enabled for execution 001 <sub>H</sub> Code Protection Range-n enabled for execution
<b>RES</b>	31:10	r	<b>Reserved</b>

### CPUx Data Protection Read Enable Register Set y

**DPRE\_y (y=0-3)**

**CPUx Data Protection Read Enable Register Set y( $1E010_H+y*4$ ) Application Reset Value: 0000 0000<sub>H</sub>**

**DPRE\_y (y=4-5)**

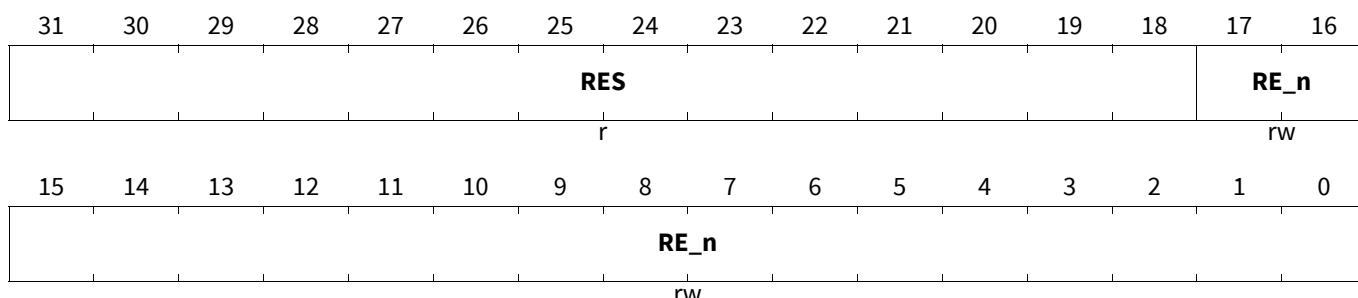
**CPUx Data Protection Read Enable Register Set y( $1E050_H+(y-4)*4$ ) Application Reset Value: 0000 0000<sub>H</sub>**

**CPU\_DPRE\_y (y=0-3)**

**Short address for domain CSFR ( $0E010_H+y*4$ ) Application Reset Value: 0000 0000<sub>H</sub>**

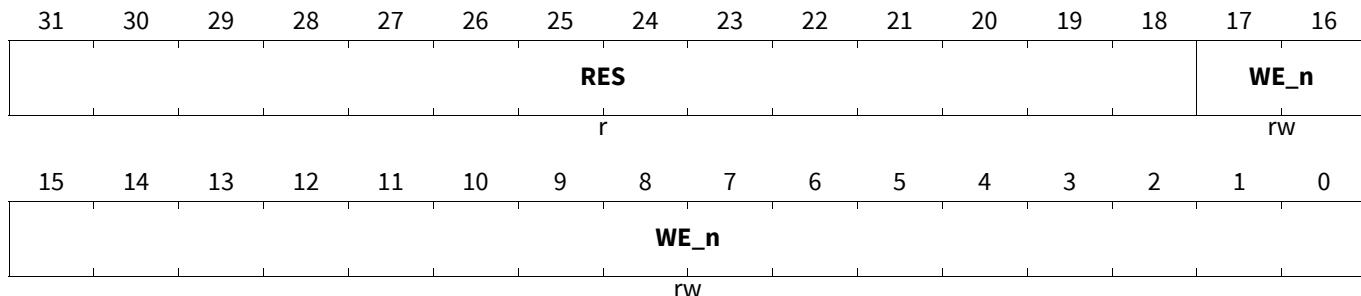
**CPU\_DPRE\_y (y=4-5)**

**Short address for domain CSFR ( $0E050_H+(y-4)*4$ ) Application Reset Value: 0000 0000<sub>H</sub>**



**CPU Subsystem**

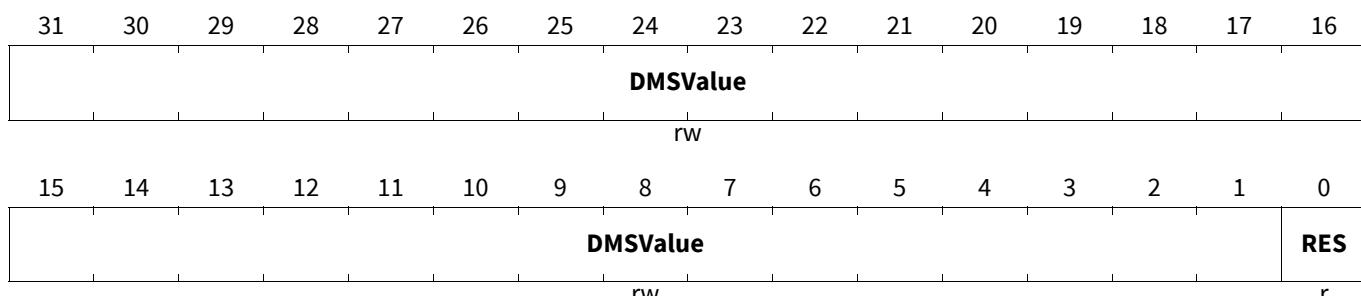
Field	Bits	Type	Description
<b>RE_n</b>	17:0	rw	<b>Read Enable Range Select - RE[n]</b> 00000 <sub>H</sub> Data Protection Range-n not enabled for data read 00001 <sub>H</sub> Data Protection Range-n enabled for data read
<b>RES</b>	31:18	r	<b>Reserved</b>

**CPUx Data Protection Write Enable Register Set y****DPWE\_y (y=0-3)****CPUx Data Protection Write Enable Register Set y(1E020<sub>H</sub>+y\*4)**      **Application Reset Value: 0000 0000<sub>H</sub>****DPWE\_y (y=4-5)****CPUx Data Protection Write Enable Register Set y(1E060<sub>H</sub>+(y-4)\*4)**      **Application Reset Value: 0000 0000<sub>H</sub>****CPU\_DPWE\_y (y=0-3)****Short address for domain CSFR (0E020<sub>H</sub>+y\*4)**      **Application Reset Value: 0000 0000<sub>H</sub>****CPU\_DPWE\_y (y=4-5)****Short address for domain CSFR (0E060<sub>H</sub>+(y-4)\*4)**      **Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>WE_n</b>	17:0	rw	<b>Write Enable Range Select - WE[n]</b> 00000 <sub>H</sub> Data Protection Range-n not enabled for data write 00001 <sub>H</sub> Data Protection Range-n enabled for data write
<b>RES</b>	31:18	r	<b>Reserved</b>

**CPUx Debug Monitor Start Address**

The DMS reset value is {20'hA0000,3'B0001,CORE\_ID,6'B000000}.

**DMS****CPUx Debug Monitor Start Address (1FD40<sub>H</sub>)**      **Debug Reset Value: A000 0XX0<sub>H</sub>****CPU\_DMS****Short address for domain CSFR (0FD40<sub>H</sub>)**      **Debug Reset Value: A000 0XX0<sub>H</sub>**

## CPU Subsystem

Field	Bits	Type	Description
<b>RES</b>	0	r	<b>Reserved</b>
<b>DMSValue</b>	31:1	rw	<b>Debug Monitor Start Address</b> The address at which monitor code execution begins when a breakpoint trap is taken.

### CPUx Debug Context Save Area Pointer

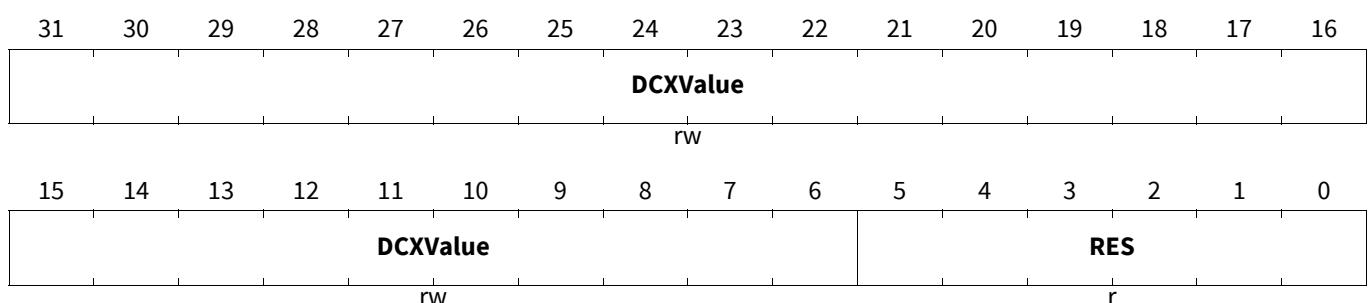
The reset value of the DCX register is {20'hA0000,3'b010,CORE\_ID,6'b000000}.

#### DCX

**CPUx Debug Context Save Area Pointer** **(1FD44<sub>H</sub>)** **Debug Reset Value: A000 0XX0<sub>H</sub>**

**CPU\_DCX**

**Short address for domain CSFR** **(0FD44<sub>H</sub>)** **Debug Reset Value: A000 0XX0<sub>H</sub>**



Field	Bits	Type	Description
<b>RES</b>	5:0	r	<b>Reserved</b>
<b>DCXValue</b>	31:6	rw	<b>Debug Context Save Area Pointer</b> Address where the debug context is stored following a breakpoint trap.

## CPU Subsystem

### 5.4 Safety Measures

The CPUs implements a number of safety concepts These are detailed in the following sections.

#### 5.4.1 SRI Bus Master Address Phase Error Injection

To allow the SRI address phase error detection system to be tested it is necessary to inject errors during the address phase of an SRI transaction. This is done by the TC1.6.2P SRI bus master for data accesses only. The SRI master selectively inverts individual bits of the address phase of an SRI data read or write phase ECC packet and is controlled using the SEGEN register. The SEGEN register contains three fields:- an enable (AE), an error descriptor (ADTYPE) and a bit flip (ADFLIP) field. When enabled for an address phase error the address ECC bits indicated by the flip field are inverted for the next SRI data read or write bus transaction performed. Following the transaction the enable bit is cleared by hardware. This mechanism allows selected bits of the SRI address ECC to be corrupted for a single transaction. The SEGEN register is CPUx ENDINIT protected.

#### 5.4.2 SRI Bus Master Write Phase Error Injection

To allow the SRI data phase error detection system to be tested it is necessary to inject errors during the data phase of an SRI write transaction. This is done by the TC1.6.2P SRI bus master. The SRI bus master selectively inverts individual bits of the SRI write phase ECC packet and is controlled using the SEGEN register. The SEGEN register contains three fields:- an enable (AE), an error descriptor (ADTYPE) and a bit flip (ADFLIP) field. When enabled for a data phase error the data phase ECC bits indicated by the flip field are inverted for the next SRI write bus transaction performed. Following the transaction the enable bit is cleared by hardware. This mechanism allows selected bits of the SRI data ECC to be corrupted for a single transaction. The SEGEN register is CPUx ENDINIT protected.

#### 5.4.3 SRI bus Slave Read Phase Error Injection

To allow the SRI data phase error detection system to be tested it is necessary to inject errors during the data phase of an SRI read transaction. This is done by the TC1.6.2P SRI bus slave interface. The slave selectively inverts individual bits of the SRI read phase ECC packet and is controlled using the SEGEN register. The SEGEN register contains three fields:- an enable (AE), an error descriptor (ADTYPE) and a bit flip (ADFLIP) field. When enabled for a data phase error the data phase ECC bits indicated by the flip field are inverted for the next SRI read bus transaction performed by the DMI. Following the transaction the enable bit is cleared by hardware. This mechanism allows selected bits of the SRI data ECC to be corrupted for a single transaction. The SEGEN register is CPUx ENDINIT protected.

#### 5.4.4 SRI Error Capture

The SRI master and slave interfaces of the CPU implement the standard SRI ECC system. Error information detected during SRI transactions at the SRI master and slave interfaces is captured in the PIETR and PIEAR, or DIETR and DIEAR registers. On detection of an error the error information is captured and the relevant IED bit is set. No further error information is captured until this bit is cleared by software.

Error conditions at the SRI program interfaces are notified to the Safety Management Unit via the pbus\_err\_o output from the CPU.

Error conditions at the SRI data interfaces are notified to the Safety Management Unit via the dbus\_err\_o output from the CPU.

If an error is detected at an SRI slave interface during a write operation then the erroneous data will not be written.

## CPU Subsystem

### 5.4.5 SRI Safe Data Master tag

In order to differentiate between data accesses from safe and regular tasks a new safe task identification bit is introduced into the PSW register (PSW.S).

An SRI data access performed by a task when the PSW.S bit is set uses the safe data master tag. When this bit is not set the regular data master tag is used for the access. There is only one program master tag. This is used for SRI program fetches by both safe and regular tasks.

The initial value of the PSW.S bit for interrupt handlers is defined by the SYSCON.IS bit. The initial value of the PSW.S bit for trap handlers is defined by the SYSCON.IT bit.

On a trap the save of the current context is performed using the data master tag defined by SYSCON.TS

On an interrupt the save of the current context is performed using the data master tag defined by SYSCON.IS

### 5.4.6 Safety Protection System

The safety protection system provides protection against illegal or unintended bus accesses to the local memory system and control registers. The system comprises of two elements:- A bus MPU to gate accesses to the local PSPR, DSPR, DLMU SRAMs and the local Pflash Bank (LPB), and an register access enable system to gate write access to the local control registers.

#### 5.4.6.1 Bus MPU

The Bus MPU comprises of:-

- Eight read and write protected regions of scratch pad memory (PSPR, DSPR) with enables for reads and writes on a per bus master basis.
- Eight read and write protected regions of DLMU with enables for reads and writes on a per bus master basis.
- Individual master read enables for accesses to the local PFlash Bank (LPB)

The protection scheme is based on the use of SRI tags to identify the master attempting the access and allows for a six bit tag individually identifying up to 64 masters.

#### General operation

A read-modify-write operation requires both read and write permissions to be enabled.

All safety protection registers are protected from modification by the safety\_endinit signal.

After reset the Safety Protection System will be enabled for access from all masters.

When altering protection settings, it should be noted that, due to access pipelining and resynchronization delays in the register block, a write to a memory address affected by the protection change occurring immediately after the register write initiating the change may, or may not, be affected by the changed settings.

Whenever a Safety Memory Protection violation occurs the event is notified to the Safety Management Unit via the tc16\_safe\_prot\_err\_o output signal from the CPU. The PIETR/DIETR and PIEAR/DIEAR registers are updated to aid localisation of the error

The following masters are allowed read access to all memories under all circumstances:- Cerburus, HSM, IOM

*Note:* Additional information can be found in the On-Chip Bus chapter under SRI Errors.

#### Scratch Pad SRAMs

Each scratchpad region is defined using the registers, SPR\_SPROT\_RGNLAI (i=0-7) to define the lower address of the region, SPR\_SPROT\_RGNUAI (x=0-7) to define the upper address of the region.

Each region may be enabled for writes on a per bus master basis using the register SPR\_SPROT\_RGNACCENAI\_W (Masters 31-0) and SPR\_SPROT\_RGNACCENBi\_W (Masters 63-32),

## CPU Subsystem

Each region may be enabled for reads on a per bus master basis using the register SPR\_SPROT\_RGNACCENAi\_R (Masters 31-0) and SPR\_SPROT\_RGNACCENBi\_R (Masters 63-32),

A write access to the PSPR/DSPR memory is seen as valid if the master tag of the access is enabled in the SPR\_SPROT\_RGNACCENi\_W register and the address of the access satisfies the following relationship:-

$\text{SPR\_SPROT\_RGNLAI} \leq \text{address} < \text{SPR\_SPROT\_RGNUAI}$

A read access to the PSPR/DSPR is seen as valid if the master tag of the access is enabled in the SPR\_SPROT\_RGNACCENi\_R register and the address of the access satisfies the following relationship:-

$\text{SPR\_SPROT\_RGNLAI} \leq \text{address} < \text{SPR\_SPROT\_RGNUAI}$

If any of these conditions are not satisfied, the access is seen as invalid.

Access from all masters to the local DSPR (excluding data access from the local CPU) are checked by the SPR safety mechanism.

Access from all masters to the local PSPR (excluding fetch access from the local CPU) are checked by the SPR safety mechanism.

## DLMU SRAMs

Each DLMU region is defined using the registers, DLMU\_SPROT\_RGNLAI (i=0-7) to define the lower address of the region, DLMU\_SPROT\_RGNUAx (i=0-7) to define the upper address of the region.

Each region may be enabled for writes on a per bus master basis using the register DLMU\_SPROT\_RGNACCENAi\_W (Masters 31-0) and DLMU\_SPROT\_RGNACCENBi\_W (Masters 63-32),

Each region may be enabled for reads on a per bus master basis using the register DLMU\_SPROT\_RGNACCENAi\_R (Masters 31-0) and DLMU\_SPROT\_RGNACCENBi\_R (Masters 63-32),

A write access to the local DLMU memory is seen as valid if the master tag of the access is enabled in the DLMU\_SPROT\_RGNACCENi\_W register and the address of the access satisfies the following relationship:-

$\text{DLMU\_SPROT\_RGNLAI} \leq \text{address} < \text{DLMU\_SPROT\_RGNUAI}$

A read access to the local DLMU memory is seen as valid if the master tag of the access is enabled in the DLMU\_SPROT\_RGNACCENi\_R register and the address of the access satisfies the following relationship:-

$\text{DLMU\_SPROT\_RGNLAI} \leq \text{address} < \text{DLMU\_SPROT\_RGNUAI}$

If any of these conditions are not satisfied, the access is seen as invalid.

Accesses from all masters to the local DLMU (including those from the local CPU) are checked by the DLMU safety protection mechanism.

## Local Pflash Bank

The local PFlash bank is protected from read accesses on a per master basis. The individual masters allowed to read the LPB are selected by the LPB\_SPROT\_ACCENA\_R (master31-0) and LPB\_SPROT\_ACCENB\_R (master 63-31) registers.

Accesses from all masters to the LPB (including those from the local CPU) are checked by the LPB safety protection mechanism.

### 5.4.6.2 Register Access Enable Protection

The CPUs implement the standard memory protection scheme for peripheral registers using the SFR\_SPROT\_ACCENA\_W (masters 31-0) and SFR\_SPROT\_ACCENB\_W (masters 63-32) register. This allows all CPU CSFR and SFR registers to be protected from write access by untrusted masters.

The SFR\_SPROT\_ACCENA\_W and SFR\_SPROT\_ACCENB\_W registers define which masters may write the SFR and CSFR registers via bus access through the SRI slave interface.

The SFR\_SPROT\_ACCENA\_W and SFR\_SPROT\_ACCENB\_W registers are protected by the safety\_endinit signal.

---

**CPU Subsystem**

In all cases the master is identified using the SRI tag of the access. See On Chip Bus chapter for the product's TAG ID to master peripheral mapping.

Whenever a Safety Register Protection violation occurs the event is notified to the Safety Management Unit via the safe\_prot\_err\_o output signal from the CPU. The PIETR and PIEAR registers are updated to aid localisation of the error.

The safety protection registers themselves are not subject to safety protection but are protected by the safety\_endinit signal.

#### **5.4.7 Registers Implementing Safety Features**

## CPU Subsystem

### 5.4.7.1 SRI safety registers

#### CPUx SRI Error Generation Register

The SEGEN register controls the injection of SRI errors from the DMI.

##### SEGEN

###### CPUx SRI Error Generation Register

( $11030_H$ )

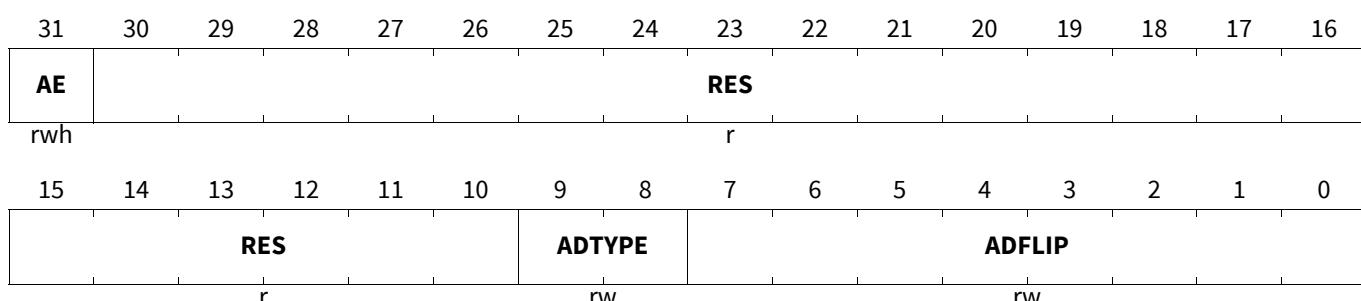
**Application Reset Value:** 0000 0000 $_H$

###### CPU\_SEGEN

###### Short address for domain CSFR

( $01030_H$ )

**Application Reset Value:** 0000 0000 $_H$



Field	Bits	Type	Description
<b>ADFLIP</b>	7:0	rw	<b>Address ECC Bit Flip</b> SRI address ECC Bits to be flipped on the next read or write transaction from the DMI when enabled by AE. 00 $_H$ No Flip 01 $_H$ Flip
<b>ADTYPE</b>	9:8	rw	<b>Type of error</b> 00 $_B$ Data Master Address Phase 01 $_B$ Data Master Write Data 10 $_B$ Data Slave Read Data 11 $_B$ Reserved
<b>RES</b>	30:10	r	<b>Reserved</b>
<b>AE</b>	31	rwh	<b>Activate Error Enable</b> Enabled the selective inverting of SRI ECC packet bits defined by ADFLIP. This bit will be cleared by hardware after the next SRI read or write transaction from the DMI. 0 $_B$ Not Enabled 1 $_B$ Enabled

## CPU Subsystem

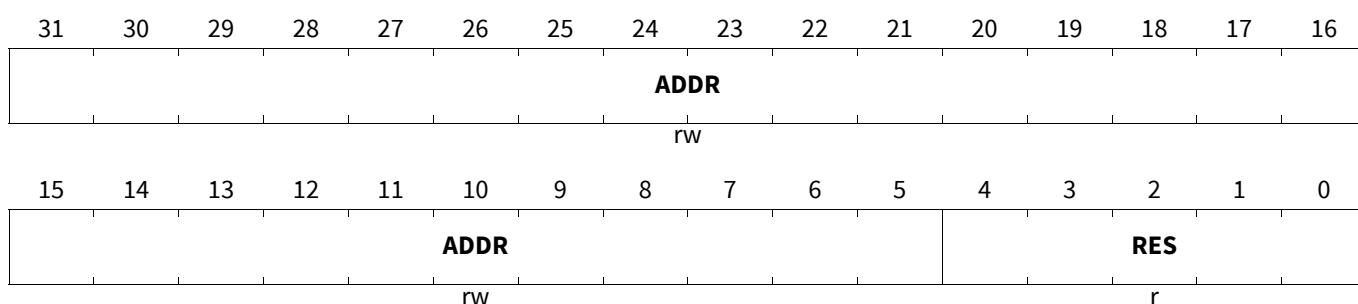
### 5.4.7.2 Safety Protection registers

#### CPUx Safety Protection SPR Region Lower Address Register i

The SPR\_SPROT\_RGNLAI defines the lower address of a region of PSPR/DSPR memory, SPR\_SPROT\_RGNUAi defines the upper address and the SPR\_SPROT\_RGNACCENi\_R/W registers define the SRI tags allowed access to the region. Address ranges can be set to be larger than the SPR SRAM address space but only accesses to the SPR SRAM are affected by these registers. The minimum resolution of the comparison logic is 32<sub>D</sub> bytes so address bits 4<sub>D</sub> down to 0<sub>D</sub> are not used

##### SPR\_SPROT\_RGNLAI (i=0-7)

**CPUx Safety Protection SPR Region Lower Address Register i(0E000<sub>H</sub>+i\*10<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**



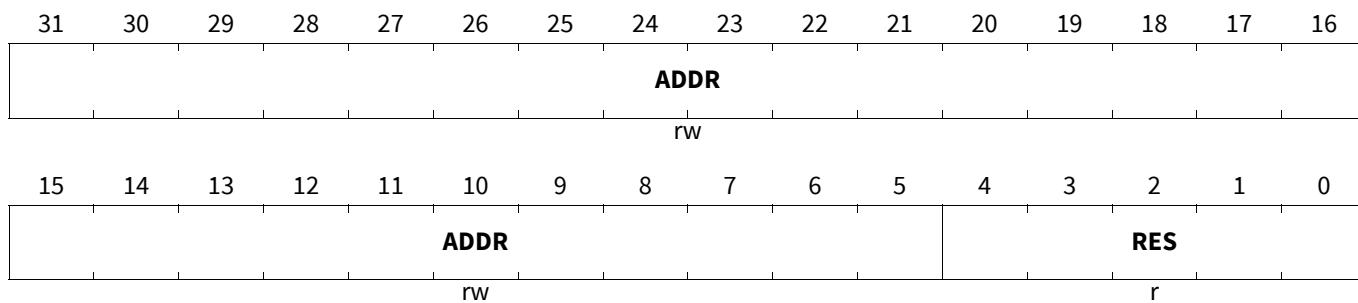
Field	Bits	Type	Description
<b>RES</b>	4:0	r	<b>Reserved</b>
<b>ADDR</b>	31:5	rw	<b>Region Lower Address</b> Bits 31 to 5 of the address which is the lower bound of the defined memory region

#### CPUx Safety Protection SPR Region Upper Address Register i

The SPR\_SPROT\_RGNUAi defines the upper address of a region of PSPR/DSPR memory. The minimum resolution of the comparison logic is 32<sub>D</sub> bytes so address bits 4<sub>D</sub> down to 0<sub>D</sub> are not used

##### SPR\_SPROT\_RGNUAi (i=0-7)

**CPUx Safety Protection SPR Region Upper Address Register i(0E004<sub>H</sub>+i\*10<sub>H</sub>) Application Reset Value: FFFF FFE0<sub>H</sub>**



Field	Bits	Type	Description
<b>RES</b>	4:0	r	<b>Reserved</b> Unused bits will always read as 0 <sub>B</sub> . Written value will be ignored

## CPU Subsystem

Field	Bits	Type	Description
<b>ADDR</b>	31:5	rw	<b>Region Upper Address</b> Bits 31 to 5 of the address which is the upper bound of the defined memory region

### CPUx Safety Protection SPR Region Write Access Enable Register Ai

The Write Access Enable Register A controls write access for transactions to the SPR safety protected memory region with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

#### SPR\_SPROT\_RGNACCENAi\_W (i=0-7)

**CPUx Safety Protection SPR Region Write Access Enable Register Ai(0E008<sub>H</sub>+i\*10<sub>H</sub>) Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN31</b>	<b>EN30</b>	<b>EN29</b>	<b>EN28</b>	<b>EN27</b>	<b>EN26</b>	<b>EN25</b>	<b>EN24</b>	<b>EN23</b>	<b>EN22</b>	<b>EN21</b>	<b>EN20</b>	<b>EN19</b>	<b>EN18</b>	<b>EN17</b>	<b>EN16</b>
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EN15</b>	<b>EN14</b>	<b>EN13</b>	<b>EN12</b>	<b>EN11</b>	<b>EN10</b>	<b>EN9</b>	<b>EN8</b>	<b>EN7</b>	<b>EN6</b>	<b>EN5</b>	<b>EN4</b>	<b>EN3</b>	<b>EN2</b>	<b>EN1</b>	<b>ENO</b>
rw															

Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n: 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

### CPUx Safety Protection SPR Region Write Access Enable Register Bi

The Write Access Enable Register B controls write access for transactions to the SPR safety protected memory region with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

#### SPR\_SPROT\_RGNACCENBi\_W (i=0-7)

**CPUx Safety Protection SPR Region Write Access Enable Register Bi(0E00C<sub>H</sub>+i\*10<sub>H</sub>) Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN63</b>	<b>EN62</b>	<b>EN61</b>	<b>EN60</b>	<b>EN59</b>	<b>EN58</b>	<b>EN57</b>	<b>EN56</b>	<b>EN55</b>	<b>EN54</b>	<b>EN53</b>	<b>EN52</b>	<b>EN51</b>	<b>EN50</b>	<b>EN49</b>	<b>EN48</b>
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EN47</b>	<b>EN46</b>	<b>EN45</b>	<b>EN44</b>	<b>EN43</b>	<b>EN42</b>	<b>EN41</b>	<b>EN40</b>	<b>EN39</b>	<b>EN38</b>	<b>EN37</b>	<b>EN36</b>	<b>EN35</b>	<b>EN34</b>	<b>EN33</b>	<b>EN32</b>
rw															

## CPU Subsystem

Field	Bits	Type	Description
<b>ENn (n=32-63)</b>	n-32	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n: 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

### CPUx Safety Protection SPR Region Read Access Enable Register Ai

The Read Access Enable Register A controls read access for transactions to the SPR safety protected memory region with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

#### SPR\_SPROT\_RGNACCENAI\_R (i=0-7)

**CPUx Safety Protection SPR Region Read Access Enable Register Ai(0E088<sub>H</sub>+i\*10<sub>H</sub>) Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables read access from the module kernel addresses for transactions with the Master TAG ID n: 0 <sub>B</sub> Read access will not be executed 1 <sub>B</sub> Read access will be executed

### CPUx Safety Protection SPR Region Read Access Enable Register Bi

The Read Access Enable Register B controls read access for transactions to the SPR safety protected memory region with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

**CPU Subsystem****SPR\_SPROT\_RGNACCENBi\_R (i=0-7)****CPUx Safety Protection SPR Region Read Access Enable Register Bi(0E08C<sub>H</sub>+i\*10<sub>H</sub>)****Application Reset****Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw															

Field	Bits	Type	Description
ENn (n=32-63)	n-32	rw	<p><b>Access Enable for Master TAG ID n</b></p> <p>This bit enables read access from the module kernel addresses for transactions with the Master TAG ID n:</p> <p>0<sub>B</sub> Read access will not be executed</p> <p>1<sub>B</sub> Read access will be executed</p>

**CPUx Safety Protection DLMU Region Lower Address Register i**

The DLMU\_SPROT\_RGNLAI defines the lower address of a region of local DLMU memory, DLMU\_SPROT\_RGNUI defines the upper address and the DLMU\_SPROT\_RGNACCENi\_R/W registers define the SRI tags allowed access to the region. Address ranges can be set to be larger than the DLMU SRAM address space but only accesses to the DLMU SRAM are affected by these registers. The minimum resolution of the comparison logic is 32<sub>D</sub> bytes so address bits 4<sub>D</sub> down to 0<sub>D</sub> are not used

**DLMU\_SPROT\_RGNLAI (i=0-7)****CPUx Safety Protection DLMU Region Lower Address Register i(0E200<sub>H</sub>+i\*10<sub>H</sub>)****Application Reset Value:****0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR															

Field	Bits	Type	Description
RES	4:0	r	<b>Reserved</b>
ADDR	31:5	rw	<p><b>Region Lower Address</b></p> <p>Bits 31 to 5 of the address which is the lower bound of the defined memory region</p>

**CPU Subsystem****CPUX Safety protection DLMU Region Upper Address Register i**

The DLMU\_SPROT\_RGNLAI defines the upper address of a region of local DLMU memory. The minimum resolution of the comparison logic is 32<sub>D</sub> bytes so address bits 4<sub>D</sub> down to 0<sub>D</sub> are not used

**DLMU\_SPROT\_RGNUAI (i=0-7)**

**CPUX Safety protection DLMU Region Upper Address Register i(0E204<sub>H</sub>+i\*10<sub>H</sub>) Application Reset Value: FFFF FFEO<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										RES					
r															

Field	Bits	Type	Description
<b>RES</b>	4:0	r	<b>Reserved</b> Unused bits will always read as 0 <sub>B</sub> . Written value will be ignored
<b>ADDR</b>	31:5	rw	<b>Region Upper Address</b> Bits 31 to 5 of the address which is the upper bound of the defined memory region

**CPUX Safety Protection Region DLMU Write Access Enable Register Ai**

The Write Access Enable Register A controls write access for transactions to the DLMU safety protected memory region with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

**DLMU\_SPROT\_RGNACCENAi\_W (i=0-7)**

**CPUX Safety Protection Region DLMU Write Access Enable Register Ai(0E208<sub>H</sub>+i\*10<sub>H</sub>) Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN31</b>	<b>EN30</b>	<b>EN29</b>	<b>EN28</b>	<b>EN27</b>	<b>EN26</b>	<b>EN25</b>	<b>EN24</b>	<b>EN23</b>	<b>EN22</b>	<b>EN21</b>	<b>EN20</b>	<b>EN19</b>	<b>EN18</b>	<b>EN17</b>	<b>EN16</b>
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EN15</b>	<b>EN14</b>	<b>EN13</b>	<b>EN12</b>	<b>EN11</b>	<b>EN10</b>	<b>EN9</b>	<b>EN8</b>	<b>EN7</b>	<b>EN6</b>	<b>EN5</b>	<b>EN4</b>	<b>EN3</b>	<b>EN2</b>	<b>EN1</b>	<b>ENO</b>
rw															

Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n: 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

**CPU Subsystem****CPUx Safety Protection Region DLMU Write Access Enable Register Bi**

The Write Access Enable Register B controls write access for transactions to the DLMU safety protected memory region with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

**DLMU\_SPROT\_RGNACCENBi\_W (i=0-7)**

**CPUx Safety Protection Region DLMU Write Access Enable Register Bi(0E20C<sub>H</sub>+i\*10<sub>H</sub>) Application Reset**

**Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw															

Field	Bits	Type	Description
ENn (n=32-63)	n-32	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n: 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

**CPUx Safety Protection Region DLMU Read Access Enable Register Ai**

The Read Access Enable Register A controls read access for transactions to the DLMU safety protected memory region with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

**DLMU\_SPROT\_RGNACCENAi\_R (i=0-7)**

**CPUx Safety Protection Region DLMU Read Access Enable Register Ai(0E288<sub>H</sub>+i\*10<sub>H</sub>) Application Reset**

**Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

## CPU Subsystem

Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables read access from the module kernel addresses for transactions with the Master TAG ID n: 0 <sub>B</sub> Read access will not be executed 1 <sub>B</sub> Read access will be executed

### CPUx Safety Protection Region DLMU Read Access Enable Register Bi

The Read Access Enable Register B controls read access for transactions to the DLMU safety protected memory region with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

#### DLMU\_SPROT\_RGNACCENBi\_R (i=0-7)

**CPUx Safety Protection Region DLMU Read Access Enable Register Bi(0E28C<sub>H</sub>+i\*10<sub>H</sub>) Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw															

Field	Bits	Type	Description
<b>ENn (n=32-63)</b>	n-32	rw	<b>Access Enable for Master TAG ID n</b> This bit enables read access from the module kernel addresses for transactions with the Master TAG ID n: 0 <sub>B</sub> Read access will not be executed 1 <sub>B</sub> Read access will be executed

### CPUx Safety Protection Region LPB Read Access Enable Register A

The Access Enable Register A controls read access for transactions to the local Pflash BanK (LPB) with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

## CPU Subsystem

### LPB\_SPROT\_ACCENA\_R

**CPUx Safety Protection Region LPB Read Access Enable Register A(0E110<sub>H</sub>) Application Reset Value: FFFF  
FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN31</b>	<b>EN30</b>	<b>EN29</b>	<b>EN28</b>	<b>EN27</b>	<b>EN26</b>	<b>EN25</b>	<b>EN24</b>	<b>EN23</b>	<b>EN22</b>	<b>EN21</b>	<b>EN20</b>	<b>EN19</b>	<b>EN18</b>	<b>EN17</b>	<b>EN16</b>
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EN15</b>	<b>EN14</b>	<b>EN13</b>	<b>EN12</b>	<b>EN11</b>	<b>EN10</b>	<b>EN9</b>	<b>EN8</b>	<b>EN7</b>	<b>EN6</b>	<b>EN5</b>	<b>EN4</b>	<b>EN3</b>	<b>EN2</b>	<b>EN1</b>	<b>ENO</b>
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables read access from the module kernel addresses for transactions with the Master TAG ID n: 0 <sub>B</sub> Read access will not be executed 1 <sub>B</sub> Read access will be executed

### CPUx Safety Protection Region LPB Read Access Enable Register B

The Access Enable Register A controls read access for transactions to the local Pflash BanK (LPB) with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). This register is unused in the current product and hence is reserved.

### LPB\_SPROT\_ACCENB\_R

**CPUx Safety Protection Region LPB Read Access Enable Register B(0E114<sub>H</sub>) Application Reset Value: FFFF  
FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN63</b>	<b>EN62</b>	<b>EN61</b>	<b>EN60</b>	<b>EN59</b>	<b>EN58</b>	<b>EN57</b>	<b>EN56</b>	<b>EN55</b>	<b>EN54</b>	<b>EN53</b>	<b>EN52</b>	<b>EN51</b>	<b>EN50</b>	<b>EN49</b>	<b>EN48</b>
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EN47</b>	<b>EN46</b>	<b>EN45</b>	<b>EN44</b>	<b>EN43</b>	<b>EN42</b>	<b>EN41</b>	<b>EN40</b>	<b>EN39</b>	<b>EN38</b>	<b>EN37</b>	<b>EN36</b>	<b>EN35</b>	<b>EN34</b>	<b>EN33</b>	<b>EN32</b>
rw															

Field	Bits	Type	Description
<b>ENn (n=32-63)</b>	n-32	rw	<b>Access Enable for Master TAG ID n</b> This bit enables read access from the module kernel addresses for transactions with the Master TAG ID n: 0 <sub>B</sub> Read access will not be executed 1 <sub>B</sub> Read access will be executed

**CPU Subsystem****CPUx Safety Protection Register Access Enable Register A**

The Access Enable Register A controls write access for transactions to local CSFR/SFR registers with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

**SFR\_SPROT\_ACCENA\_W**

**CPUx Safety Protection Register Access Enable Register A(0E100<sub>H</sub>) Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

Field	Bits	Type	Description
ENn (n=0-31)	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n: 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

**CPUx Safety Protection Region Access Enable Register B**

The Access Enable Register B controls write access for transactions to CSFR/SFR registers with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

**SFR\_SPROT\_ACCENB\_W**

**CPUx Safety Protection Region Access Enable Register B(0E104<sub>H</sub>) Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw															

Field	Bits	Type	Description
ENn (n=32-63)	n-32	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n: 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

## CPU Subsystem

### 5.5 IO Interfaces

Following tables describe the interfaces for this module.

### 5.6 Revision History

**Table 132 Revision History**

Reference	Change to Previous Version	Comment
<b>V1.1.16</b>		
<a href="#">Page 125</a>	Added a note to indicate that additional information regarding bus behaviour can be found in the On-Chip bus chapter.	
<a href="#">Page 115</a>	Clarification of the section on CPU architecture registers in order to provide the implementation specific reset values and some descriptions. The information was missing in previous versions of the document. The following revision history items provide additional details on the changes.	
<a href="#">Page 31</a>	Added description and reset value of Data GPRs (missing in previous version of the manual)	
<a href="#">Page 31</a>	Added description and reset value of Address GPRs (missing in previous version of the manual)	
<a href="#">Page 33</a>	Added description and reset value of FPU_TRAP_CON (missing in previous version of the manual)	
<a href="#">Page 34</a>	Added description and reset value of FPU_TRAP_OPC (missing in previous version of the manual)	
<a href="#">Page 35</a>	Added description and reset value of FPU_TRAP_SRC1 (missing in previous version of the manual)	
<a href="#">Page 35</a>	Added description and reset value of FPU_TRAP_SRC2 (missing in previous version of the manual)	
<a href="#">Page 36</a>	Added description and reset value of FPU_TRAP_SRC3 (missing in previous version of the manual)	
<a href="#">Page 38</a>	Added description and reset value of TPS_CON (missing in previous version of the manual)	
<a href="#">Page 39</a>	Added description and reset value of TPS_TIMERy (missing in previous version of the manual)	
<a href="#">Page 115</a>	Removed references to a number of duplicated registers already defined in other chapters : SMACON, TPS_CON, TPS_TIMERy, FPU_TRAP_CON, FPU_TRAP_OPC, FPU_TRAP_SRC1, FPU_TRAP_SRC2, FPU_TRAP_SRC3, PIEAR, PIETR, SFR_SPROT_ACCENA_W, SFR_SPROT_ACCENB_W	
<a href="#">Page 115</a>	Added a new subsection for registers fully defined in the architecture manual	
<a href="#">Page 115</a>	Added clarification on where to find the Program Counter reset value	
<a href="#">Page 116</a>	Added a new subsection for registers which are normally defined in the architecture manual with implementation specific reset values	

**CPU Subsystem****Table 132 Revision History (cont'd)**

<b>Reference</b>	<b>Change to Previous Version</b>	<b>Comment</b>
<a href="#">Page 116</a>	Added description and reset value of BIV (missing in previous version of the manual)	
<a href="#">Page 117</a>	Added description and reset value of BTV (missing in previous version of the manual)	
<a href="#">Page 117</a>	Added description and reset value of ISP (missing in previous version of the manual)	
<a href="#">Page 118</a>	Added description and reset value of FCX (missing in previous version of the manual)	
<a href="#">Page 118</a>	Added description and reset value of LCX (missing in previous version of the manual)	
<a href="#">Page 119</a>	Added description and reset value of Data protection ranges Lower bound registers (missing in previous version of the manual)	
<a href="#">Page 119</a>	Added description and reset value of Data protection ranges Upper bound registers (missing in previous version of the manual)	
<a href="#">Page 120</a>	Added description and reset value of Code protection ranges Lower bound registers (missing in previous version of the manual)	
<a href="#">Page 120</a>	Added description and reset value of Code protection ranges Upper bound registers (missing in previous version of the manual)	
<a href="#">Page 121</a>	Added description and reset value of Code execute protection set registers (missing in previous version of the manual)	
<a href="#">Page 121</a>	Added description and reset value of Data read protection set registers (missing in previous version of the manual)	
<a href="#">Page 122</a>	Added description and reset value of Data write protection set registers (missing in previous version of the manual)	
<a href="#">Page 122</a>	Added description and reset value of DMS (missing in previous version of the manual)	
<a href="#">Page 123</a>	Added description and reset value of DCX (missing in previous version of the manual)	
<b>V1.1.17</b>		
<a href="#">Page 18</a>	Corrected typo. Plural was intended.	
<a href="#">Page 19</a>	Replaced “system_endint” with “system registers ENDINIT” for KRSTCLR	
<a href="#">Page 19</a>	Replaced “system_endint” with “system registers ENDINIT” KRST0	
<a href="#">Page 20</a>	Replaced “system_endint” with “system registers ENDINIT” KRST1	
<a href="#">Page 75</a>	Added clarification for usage of CACHEx.x instructions with non cacheable addresses.	
<b>V1.1.18</b>		
<a href="#">Page 9</a>	Renamed TC38xEXT into TC3Ex	
<a href="#">Page 10</a>	Added new memory configuration for derivative TC3Ax	
<a href="#">Page 17</a>	Renamed SPB into SRI to remove duplication. Both columns were incorrectly stating SPB.	

**CPU Subsystem****Table 132 Revision History (cont'd)**

<b>Reference</b>	<b>Change to Previous Version</b>	<b>Comment</b>
<a href="#">Page 75</a>	Reformatted CUS_ID table mapping to accomodate more derivatives. Merged columnss based on number of CPU's per derivative. Renamed TC38xEXT into TC3Ex, TC33xED into TC33xEXT and included mention of TC3Ax derivative.	
<a href="#">Page 106</a>	Used subscript to refer to hexadecimal values in overlay OLDA section	
<a href="#">Page 107</a>	Used subscript to refer to hexadecimal values in overlay Local Memory section. Updated target segment to the correct values of $A_H$ and $8_H$	
<a href="#">Page 107</a>	Used subscript to refer to hexadecimal values in overlay external memory section	
<a href="#">Page 107</a>	Used subscript to refer to hexadecimal values in overlay DSPR & PSPR section	
<a href="#">Page 124</a>	Renamed “Safety Monitor Module” into “Safety Management Unit”	
<a href="#">Page 130</a>	Changed bitfield description of SPR_SPROT_RGNACCENAi_W register to provide a more consistent description matching similar registers in other modules. The bitfield is split into 32 unique bits clearly indicating that each bit corresponds to one Master TAG ID	
<a href="#">Page 130</a>	Same change for register SPR_SPROT_RGNACCENBi_W.	
<a href="#">Page 131</a>	Same change for register SPR_SPROT_RGNACCENAx_R.	
<a href="#">Page 131</a>	Same change for register SPR_SPROT_RGNACCENBx_R.	
<a href="#">Page 133</a>	Same change for register DLMU_SPROT_RGNACCENAx_W.	
<a href="#">Page 134</a>	Same change for register DLMU_SPROT_RGNACCENBx_W.	
<a href="#">Page 134</a>	Same change for register DLMU_SPROT_RGNACCENAx_R.	
<a href="#">Page 135</a>	Same change for register DLMU_SPROT_RGNACCENBx_R.	
<a href="#">Page 135</a>	Same change for register LPB_SPROT_ACCENA_R.	
<a href="#">Page 136</a>	Same change for register LPB_SPROT_ACCENB_R.	
<a href="#">Page 137</a>	Same change for register SFR_SPROT_ACCENA_W.	
<a href="#">Page 137</a>	Same change for register SFR_SPROT_ACCENA_W.	
<b>V1.1.19</b>		
<a href="#">Page 10</a>	Updated memory configuration of CPU2 for derivative TC3Ax. Increase DPSR size from 96KB to 240KB.	
<a href="#">Page 12</a>	Added clarification regarding store buffer operation. The previous description was errouneous in the case of non local memories and was only taking in account single core operation.	
<a href="#">Page 59</a>	Changed reset type of DMS and DCX registers from “application reset” to “debug reset” (summary table)	
<a href="#">Page 75</a>	Added table documenting old CUS_ID numbering scheme found in early samples	
<a href="#">Page 122</a>	Changed reset type of DMS register from “application reset” to “debug reset”	
<a href="#">Page 123</a>	Changed reset type of DCX register from “application reset” to “debug reset”	

**CPU Subsystem****Table 132 Revision History (cont'd)**

<b>Reference</b>	<b>Change to Previous Version</b>	<b>Comment</b>
<a href="#">Page 125</a>	Clarified operation of SPR safety protection. Added indication that an access not satisfying the access conditions is invalid.	
<a href="#">Page 126</a>	Clarified operation of DLMU safety protection. Added indication that an access not satisfying the access conditions is invalid.	
<b>V1.1.20</b>		
<a href="#">Page 61</a>	Change index variable from 'x' to intended 'i' for registers SPR_SPROT_RGNACCENAi_R and SPR_SPROT_RGNACCENBi_R to remove confusion with CPU instance variable.	
<a href="#">Page 61</a>	Change index variable from 'x' to intended 'i' for registers all DLMU_SPROT registers to remove confusion with CPU instance variable.	
<a href="#">Page 125</a>	Change index variable from 'x' to intended 'i' for registers SPR_SPROT registers to remove confusion with CPU instance variable.	
<a href="#">Page 126</a>	Change index variable from 'x' to intended 'i' for registers all DLMU_SPROT registers to remove confusion with CPU instance variable.	
<a href="#">Page 126</a>	Corrected misleading write protection description to use the DLMU_SPROT_RGNACCENi_W and DLMU_SPROT_RGNACCENBi_W instead of the SPR_SPROT.	
<a href="#">Page 126</a>	Explicitely mentioned SFR_SPROT_ACCENA_W and SFR_SPROT_ACCENB_W registers rather than SFR_SPROT_ACCENx_W which could have been misleading.	
<a href="#">Page 131</a>	Changed index variable from 'x' to 'i' for register SPR_SPROT_RGNACCENAi_R	
<a href="#">Page 131</a>	Changed index variable from 'x' to 'i' for register SPR_SPROT_RGNACCENBi_R	
<a href="#">Page 132</a>	Changed index variable from 'x' to 'i' for register DLMU_SPROT_RGNLAI	
<a href="#">Page 133</a>	Changed index variable from 'x' to 'i' for register DLMU_SPROT_RGNUAI	
<a href="#">Page 133</a>	Changed index variable from 'x' to 'i' for register DLMU_SPROT_RGNACCENAi_W	
<a href="#">Page 134</a>	Changed index variable from 'x' to 'i' for register DLMU_SPROT_RGNACCENBi_W	
<a href="#">Page 134</a>	Changed index variable from 'x' to 'i' for register DLMU_SPROT_RGNACCENAi_R	
<a href="#">Page 135</a>	Changed index variable from 'x' to 'i' for register DLMU_SPROT_RGNACCENBi_R	

---

## Non Volatile Memory (NVM) Subsystem

# 6 Non Volatile Memory (NVM) Subsystem

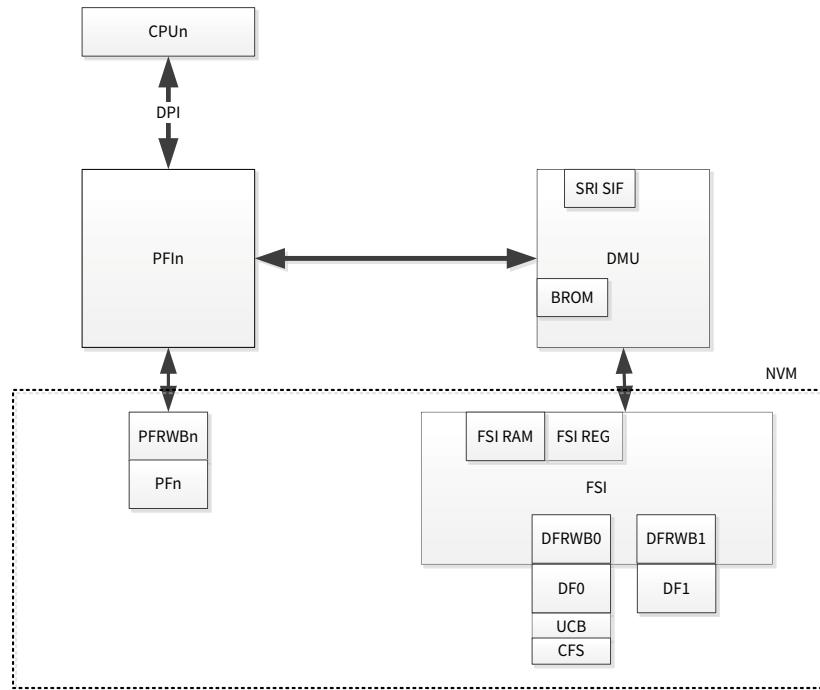
## 6.1 Overview

The Non Volatile Memory (NVM) Subsystem comprises of the Data Memory Unit (DMU), Program Flash Interface (PFI), and Non Volatile Memory module (comprising of the Flash Standard Interface (FSI), Program and Data Flash memories and Program Flash Read Write buffer (PFRWB)).

- Data Memory Unit (DMU): Controls command sequences executed on all program and data flash memories.
- Flash Standard Interface (FSI): Executes erase, program and verify operations on all flash memories.
- Program Flash (PFLASH): Divided into one or more banks each connected to a CPU. It is used by the application to store program code and data constants. Compute performance is optimized by using a point-to-point interface to minimize latency and maximize bandwidth. Each PFLASH is connected to a PFlash Read Write Buffer (PFRWB) that performs the ECC correction and detection and provides the read data to the system.
- Program Flash Interface (PFI): Each PFLASH bank has a unique point-to-point fast connection to a CPU provided by a PFI. The PFI interfaces between the CPU and the PFRWB and contains the Prefetch Buffers for storing speculative data.
- Data Flash (DFLASH): The Data Flash Module is used to emulate EEPROM and store data and divided into two banks. DFLASH read accesses are relatively slow compared to PFLASH accesses. The DFlash Read Write Buffer (DFRWB) in the FSI interfaces to the DFLASH to provide the read data. Data Flash Module also contains regions to store configuration data - User Configuration Blocks (UCBs), and Configuration Sector (CFS) which is not accessible by user.
- Boot ROM (BROM): Connected to the system via the DMU SRI port.
  - Tuning protection (commonly called the “Secure Watchdog”) to protect user software and data from malfunctions.

**Attention:** *The ‘Non Volatile Memory Subsystem’ chapter is the AURIX PMU chapter re-structured for closer alignment to AURIXTC3XX product architecture. It comprises of the DMU, PFI, NVM and UCB Chapters. Please note that the application accessible registers located in the FSI, and the PFLASH read status and control registers are described in the NVM chapter.*

## Non Volatile Memory (NVM) Subsystem



**Figure 57 Non Volatile Memory (NVM) Subsystem**

The purpose of the PFLASH NVM is:

- One or more PFLASH banks stores program code and data constants.
- Implementation of Erase Counters.

The purpose of the DFLASH NVM is:

- Emulation of Electrically Erasable Programmable Read Only Memory (EEPROM):
  - CPU-EEPROM used by the user application.
  - HSM-EEPROM used by the security application.
- Multiple User Configuration Blocks (UCB) used for:
  - Password based read protection combined with write protection.
  - Read-only UCB configured by IFX with unique chip identifier and trimming data.
- Configuration Sector (CFS) stores system set-up data not accessible by the user.

Data stored in the NVM is protected by ECC checksum.

- An ECC decoder at the output of the NVM corrects and detects faults in the NVM array.
- The NVM is fault tolerant and supports system operation in the presence of a number of NVM bit errors.
- For Program Flash the calculation of the ECC checksum is extended across the address to provide read protection against addressing faults.

If the Flash is not operating in the application then the NVM may be programmed and erased by command sequences executed by the FSI micro controller. All read accesses to Flash are memory mapped reads. Margin read levels may be used to check how completely a cell is programmed or erased.

The Non Volatile Memory interface micro architecture includes a security layer and a safety layer.

### Security Layer (provided by DMU and PFI)

- Read protection is enabled/disabled globally for the whole Flash Module.

## Non Volatile Memory (NVM) Subsystem

- Write protection is enabled/disabled with a Flash Module sector based granularity.

### Safety Layer

- Master specific read access protection to each Flash Module (Bank).
- Master specific read and write access control to individual Special Function Registers (SFRs).
- Integrity of data stored in the NVM is ensured by an ECC checksum
- Integrity of PFlash read path is ensured by monitoring of read parameters in the FSI (MISR, redundant Flip Flops etc.), PFI partial lockstep mechanism, protection of PFlash wait cycles with ECC checksum, protection of data from PFI to CPU by ECC checksum and an additional safety mechanism to ensure that the local PFlash is not being programmed/erased when not expected by PFI.

## 6.2 Functional Description

### 6.2.1 Definition of Terms

The Non Volatile Memory modules use the following terminology:

#### Flash Operation Terms

- **Erasing:** The erased state of a Flash cell is logical ‘0’. Forcing a cell to this state is called “erasing”. Complete Flash sectors are erased. All Flash cells in this area incur one “cycle” that counts for the “endurance”.
- **Programming:** The programmed state of a Flash cell is logical ‘1’. Changing an erased Flash cell to this state is called “programming”. Programming bits within a page to a logic ‘1’ occurs concurrently.
- **Retention:** This is the time during which the data of a flash cell can be read reliably. The retention time is a statistical figure that depends on the operating conditions of the device (e.g. temperature profile) and is affected by operations on other Flash cells in the same wordline and physical sector. With an increasing number of program/erase cycles (see endurance) the retention is lowered.
- **Endurance:** As described above the data retention is reduced with an increasing number of program/erase cycles. The maximum number of program/erase cycles of each Flash cell is called “endurance”. As for retention it is a statistical figure that depends on operating conditions, the use of the flash cells and the required quality level.

#### Flash Structure Terms

- **Flash Module:** A module that contains a Non Volatile Memory (NVM) with its own digital control logic.
- **Bank:** A “Flash module” contains separate “banks”. In the PFLASH there are one or more PFp banks and in the DFLASH there are one or more banks. “Banks” support concurrent operations with some limitations due to common logic.
- **Non Volatile Memory (NVM):** The physical memory used to store information. It is split into sectors:
  - Physical Sectors: One physical area of memory is isolated from another area of memory.
  - Logical Sectors: A physical sector is further separated into logical sectors. A logical sector is a group of wordlines for PFLASH and DFLASH, and a single wordline for UCB. A logical sector can be erased with a single operation. The plain term “sector” means “logical sector”.
  - Mini Sectors: A group of logical sectors in UCB.
- **Page:** A page is an aligned group of data double words plus an ECC extension. It is the smallest unit that can be programmed.

## Non Volatile Memory (NVM) Subsystem

- PFLASH: 4 data double words (32 bytes) plus 22-bit ECC extension.
- DFLASH: 1 data double word (8 bytes) plus 22-bit ECC extension.
- **Wordline:** An aligned group of bytes:
  - PFLASH: 1024 bytes.
  - DFLASH: 512 bytes in single ended mode and 256 bytes in complement sensing mode.
- **Program Burst:** The maximum amount of data that can be programmed with one command. The programming throughput is higher than for programming single pages.
  - PFLASH: 8 pages (256 bytes).
  - DFLASH: 4 pages (32 bytes).
- **Flash Interface:** The digital control logic to control read accesses and program and erase operations to the NVM.

### 6.2.2 Major changes from Aurix to AURIXTC3XX

- Dedicated performance interface for PFLASH read (PFI)
- New Flash structure and Flash sizes
- HSM PCODE sectors in PF0 increased
- Configuration Sector (CFS) moved to DFLASH
- Erase Counter moved to the respective PFLASH
- All Flash banks, CFS, UCB, Erase Counters and registers have separate system addresses
- Complement Sensing mode for DFLASH0/1 EEPROM
- New UCBs and Dual UCB concept
- Protection configuration in UCB updated to match new Flash structure
- New command sequences
- New power mode - Cranking mode
- New registers, and re-organisation of existing registers to accomodate architecture changes
- No legacy ECC for PFLASH, only safe ECC
- Requested Read feature removed
- DFLASH1 register access protection - change in definition of access term 'H'
- Support for Software update Over the Air (SOTA)
  - Enabling HSM PCODE sector configuration in more than one PFLASH
  - Individual PFLASH bank protection disable
  - Disabling Safety Endinit protection for PFLASH banks not used for code execution by the running application.
- High and Low priority Erase Counter regions
- New functional safety features

### 6.2.3 Flash Structure

The NVM contains the following Flash banks. The offset address of each sector is relative to the base address of its bank which is given in the device Memory Map.

## Non Volatile Memory (NVM) Subsystem

### 6.2.3.1 Program Flash Banks

All PFLASH Banks PFp are based on the same sector structure.

PFp banks may vary in size and implement the following logical sector structure:

- 3 Mbyte Program Flash Bank PFp implements logical sectors S0 to S191.
- 2 Mbyte Program Flash Bank PFp implements logical sectors S0 to S127.
- 1 Mbyte Program Flash Bank PFp implements logical sectors S0 to S63.

**Table 133 Sector Structure of PFp**

Logical Sector	Physical Sector	Offset Address	Size	Total
S0	PS0	00'0000 <sub>H</sub>	16 Kbyte	16 Kbyte
S1		00'4000 <sub>H</sub>	16 Kbyte	32 Kbyte
S2		00'8000 <sub>H</sub>	16 Kbyte	48 Kbyte
...		...	...	...
S61		0F'4000 <sub>H</sub>	16 Kbyte	992 Kbyte
S62		0F'8000 <sub>H</sub>	16 Kbyte	1008 Kbyte
S63		0F'C000 <sub>H</sub>	16 Kbyte	1024 Kbyte
S64	PS1	10'0000 <sub>H</sub>	16 Kbyte	1040 Kbyte
S65		10'4000 <sub>H</sub>	16 Kbyte	1056 Kbyte
S66		10'8000 <sub>H</sub>	16 Kbyte	1072 Kbyte
...		...	...	...
S125		1F'4000 <sub>H</sub>	16 Kbyte	2016 Kbyte
S126		1F'8000 <sub>H</sub>	16 Kbyte	2032 Kbyte
S127		1F'C000 <sub>H</sub>	16 Kbyte	2048 Kbyte
S128	PS2	20'0000 <sub>H</sub>	16 Kbyte	2064 Kbyte
S129		20'4000 <sub>H</sub>	16 Kbyte	2080 Kbyte
S130		20'8000 <sub>H</sub>	16 Kbyte	2096 Kbyte
...		...	...	...
S189		2F'4000 <sub>H</sub>	16 Kbyte	3040 Kbyte
S190		2F'8000 <sub>H</sub>	16 Kbyte	3056 Kbyte
S191		2F'C000 <sub>H</sub>	16 Kbyte	3072 Kbyte

#### 6.2.3.1.1 PFLASH Tuning Protection (TP) and HSM Support

AURIXTC3XX must support simultaneous TP and HSM operation. The lower logical sectors of PFLASH bank 0 (at [Figure 58](#)) may be configured to support TP and HSM Program Code (PCODE).

## Non Volatile Memory (NVM) Subsystem

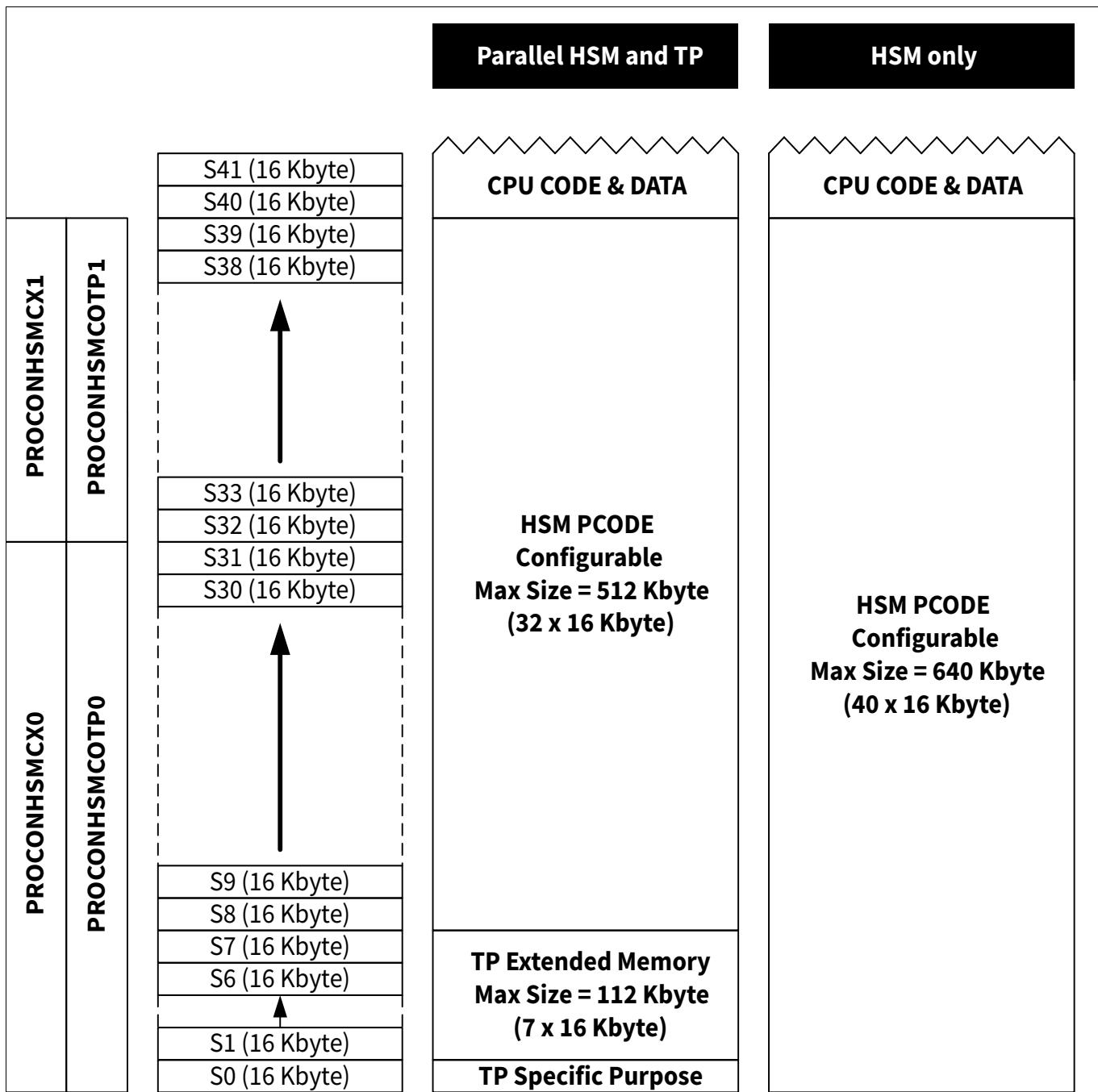


Figure 58 PFLASH TP and HSM PCODE Configuration.

### PFLASH TP Configuration

If TP is enabled (DMU\_HF\_PROCONTP.TP = 1<sub>B</sub>) then PFLASH PF0 S0 has a specific TP purpose and S0 must not be configured as “HSM\_exclusive”.

The PFLASH logical sectors above PF0 S0 may be configured to implement TP extended memory.

In addition, another PFLASH bank can have S0 configured for TP and other sectors configured to implement TP extended memory depending on the DMU\_HF\_PROCONTP.SWAPEN bit. This secondary PFLASH bank number is device dependent, and is the PFLASH bank with the lowest system address in the alternate address map provided in the MEMMAP chapter.

## Non Volatile Memory (NVM) Subsystem

### PFLASH HSM PCODE Configuration

PFLASH PF0 logical sectors S0 to S39 may be configured as “HSM\_exclusive” to support the placement of HSM PCODE at S0 or above TP sectors. HSM PCODE sectors may be configured as HSM locked forever.

### Configuration Options

If parallel TP and HSM operation are required then PF0 S0 to S39 may be configured for TP and HSM PCODE as follows:

- PF0 S0: specific TP purpose.
- PF0 S1 to S7: TP extended memory
- PF0 S8 to S39: HSM PCODE

TP, HSM PCODE and CPU address ranges must be contiguous.

### Multiple PFLASH TP and HSM PCODE configuration

In order to support “Software update Over the Air (SOTA)”, another PFLASH bank can have sectors S0 to S39 configured for TP or HSM PCODE, in addition to PFLASH0. This secondary PFLASH bank number is device dependent, and is the PFLASH bank with the lowest system address in the alternate address map provided in the MEMMAP chapter.

The same registers, DMU\_SP\_PROCONHSMCX0-1 and DMU\_SP\_PROCONHSMCOTP0-1 are used to configure the sectors in this secondary PFLASH used for HSM PCODE. The sectors not used for HSM PCODE in the secondary PFLASH bank, can be used for TP similar to PF0 and protected by enabling sectorwise protection in DMU\_HP\_PROCONOTP/DMU\_HP\_PROCONWOP/DMU\_HP\_PROCONP registers of the secondary PFLASH bank.

#### 6.2.3.1.2 Erase Counters

An Erase Counter(EC) is a dedicated 16 Kbyte logical sector within each PFLASH(PFp) bank. The erase counter may be accessed by every bus master and is read-only in the application.

##### Erase Logging

For every call of an “Erase Logical Sector Range” command sequence the FSI automatically records the logical sector start address (this is not the system address, but the address of the selected local PFLASH bank) and number of logical sectors in a 256-bit log in the corresponding Erase Counter (ECp).

**Table 134 256-bit Erase Counter Logging Entry Structure**

Bits	Description
[31:0]	Marker.
[63:32]	Start address of the first PFp logical sector.
[71:64]	Number of logical sectors erased.
[255:72]	Reserved.

The ECp 16 Kbyte erase counter supports the recording of 2 X 255 X 256-bit log entries. The Erase Counter area in each PFLASH is divided into two areas supporting 256 entries each - of which the first entry is pre-programmed and the remaining 255 entries are used to record erase operations. The first 256 entries from the Erase Counter base address is the Low priority area and the remaining is the High priority area. The user can assign priority to a logical sector in the PFLASH using the UCB\_ECPRI0. When an erase encompassing at least one high priority logical sector is triggered, it is recorded in the high priority area of the Erase Counter. Otherwise, the erase is recorded in the low priority area. The first entry of each area is a pre-programmed ECC valid entry, and cannot be used for Erase Counter entries. Each EC entry (including the first pre-programmed entry) has a 4 byte log marker.

## Non Volatile Memory (NVM) Subsystem

The marker is used to indicate the last valid EC entry. If the marker value of the entry is not equal to 0x00000000, the next EC entry is valid and can be read. If the marker value of entry is equal to 0x00000000, the last valid EC entry is reached. Thus, the next entry on address is empty (erased state) and generates ECC MBE fails when read.

### Erase Counter access

Erase counters can be read by any bus master that has the rights to read the associated PFLASH bank. All reads are performed by the local CPU through the connected PFI. All other flash operations on the Erase Counter regions fail with an SQER (See DMU Chapter for more details).

### 6.2.3.2 EEPROM Emulation with DFLASH

The term “EEPROM Emulation” designates an algorithm with the following features:

- It increases the effective endurance by spreading the EEPROM write accesses over a larger range of Flash memory.
- It ensures that all Flash cells incur a similar number of cycles independent of the update frequency of the EEPROM data (“wear levelling”).
- It manages the allocation of EEPROM data to Flash ranges so that stale data can be erased.

### Aborts and Startup

A specific requirement for the EEPROM emulation algorithm is resistance against aborts during its operation. After startup the device must be able to recover the active data without data loss and resume operation.

#### 6.2.3.2.1 DFLASH Emulation Modes

The DFLASH supports multi sector EEPROM emulation.

The DFLASH banks DFLASH0 and DFLASH1 must be operated in the same mode (i.e. single ended or complement sensing).

#### 6.2.3.2.2 Robust EEPROM Emulation

A key requirement for an EEPROM emulation algorithm is the reliability of the stored data. The DFLASH with its TECQED ECC algorithm protects perfectly against bit or bit-line oriented failures.

The ECC mechanism does not protect against wordline oriented failures. Wordline failure modes may result in a complete wordline which can no longer be programmed or erased and the data within the failed wordline may not be able to be read correctly. Wordline failures are most likely to occur during the high voltage conditions present during programming or erase operations. A robust EEPROM emulation algorithm must tolerate wordline failures and must protect against data loss in case the wordline fails during an erase or programming operation.

The following steps should be followed to achieve the necessary robustness:

- Before programming a page save the content of all other pages on the same wordline that contain active data to SRAM.
- Program the new page and compare the content of this page and of the saved pages with their reference data. This can be done with normal read margins. Ignore correctable bit-errors.
- If the data comparison fails or the programming returned with PVER error program this page and the saved content of the other pages to a different wordline.
- This procedure can be repeated if the data comparison fails again. The number of repetitions should be limited (e.g. to 3) in case the programming fails because of out-of-spec operating conditions.

## Non Volatile Memory (NVM) Subsystem

- Wordline oriented fails can also have the effect that the affected wordlines can not be erased anymore, yet the contents of the wordline data could still be read without any error indication. Sequence counters for the data blocks or other means must be used to identify old data blocks in wordlines with this type of failure mode.

Due to the specificity of each application the appropriate usage and implementation of these measures must be chosen according to the context of the application.

### 6.2.3.3 Data Flash Bank DFLASH0

DFLASH0 includes DFLASH0\_EEPROM (single ended mode at [Table 135](#) and complement sensing mode at [Table 136](#)), DFLASH0\_UCB (at [Table 137](#)) and DFLASH0\_CFS (at [Table 138](#)).

#### DFLASH0\_EEPROM Erase Operations and Erase Disturb

The number of erase operations over lifetime is limited by the parameter  $N_{ERDO}$  ( $N_{ERDOS}$  for single ended and  $N_{ERDOC}$  for complement sensing). All sectors must be used round-robin in order to prevent the accumulation of erase disturbs in static sectors<sup>1)</sup>. In derivatives with a reduced amount of DFLASH all sectors in this reduced DFLASH range must be used round-robin.

**Table 135 Sector Structure of DFLASH0\_EEPROM in Single Ended Mode**

Logical Sector	Physical Sector	Offset Address <sup>1)</sup>	Size	Total
EEPROM0	DFLASH0_EEPROM	00'0000 <sub>H</sub>	4 Kbyte	4 Kbyte
EEPROM1		00'1000 <sub>H</sub>	4 Kbyte	8 Kbyte
...		...	...	...
EEPROM14		00'E000 <sub>H</sub>	4 Kbyte	60 Kbyte
EEPROM15		00'F000 <sub>H</sub>	4 Kbyte	64 Kbyte
...		...	...	...
EEPROM22		01'6000 <sub>H</sub>	4 Kbyte	92 Kbyte
EEPROM23		01'7000 <sub>H</sub>	4 Kbyte	96 Kbyte
...		...	...	...
EEPROM30		01'E000 <sub>H</sub>	4 Kbyte	124 Kbyte
EEPROM31		01'F000 <sub>H</sub>	4 Kbyte	128 Kbyte
...		...	...	...
EEPROM62		03'E000 <sub>H</sub>	4 Kbyte	252 Kbyte
EEPROM63		03'F000 <sub>H</sub>	4 Kbyte	256 Kbyte
...		...	...	...
EEPROM126		07'E000 <sub>H</sub>	4 Kbyte	508 Kbyte
EEPROM127		07'F000 <sub>H</sub>	4 Kbyte	512 Kbyte
...		...	...	...
EEPROM254		0F'E000 <sub>H</sub>	4 Kbyte	1020 Kbyte
EEPROM255		0F'F000 <sub>H</sub>	4 Kbyte	1024 Kbyte

1) System Address = DFLASH0 Base Address (see Memory Map) + Offset Address

1) The parameter  $N_{DFD}$  (see Data Sheet) documents the erase disturb limit. It is chosen higher than needed for pure round-robin usage to cover also error cases (e.g. repetition of erase commands on the same range due to power failures during operation).

## Non Volatile Memory (NVM) Subsystem

**Attention:** For DFLASH0\_EEPROM configured in single ended mode the minimum erase size is 4 Kbyte aligned to the logical sector address boundary.

**Table 136 Sector Structure of DFLASH0\_EEPROM in Complement Sensing Mode**

Logical Sector	Physical Sector	Offset Address <sup>1)</sup>	Size	Total
EEPROM0	DFLASH0_EEPROM	00'0000 <sub>H</sub>	2 Kbyte	2 Kbyte
EEPROM1		00'0800 <sub>H</sub>	2 Kbyte	4 Kbyte
...		...	...	...
EEPROM14		00'7000 <sub>H</sub>	2 Kbyte	30 Kbyte
EEPROM15		00'7800 <sub>H</sub>	2 Kbyte	32 Kbyte
...		...	...	...
EEPROM30		00'F000 <sub>H</sub>	2 Kbyte	62 Kbyte
EEPROM31		00'F800 <sub>H</sub>	2 Kbyte	64 Kbyte
...		...	...	...
EEPROM62		01'F000 <sub>H</sub>	2 Kbyte	126 Kbyte
EEPROM63		01'F800 <sub>H</sub>	2 Kbyte	128 Kbyte
...		...	...	...
EEPROM126		03'F000 <sub>H</sub>	2 Kbyte	254 Kbyte
EEPROM127		03'F800 <sub>H</sub>	2 Kbyte	256 Kbyte
...		...	...	...
EEPROM254		07'F000 <sub>H</sub>	2 Kbyte	510 Kbyte
EEPROM255		07'F800 <sub>H</sub>	2 Kbyte	512 Kbyte

1) System Address = DFLASH0 Base Address (see Memory Map) + Offset Address

**Attention:** For DFLASH0\_EEPROM configured in complement sensing mode the minimum erase size is 2 Kbyte aligned to the logical sector address boundary.

**Table 137 Sector Structure of DFLASH0\_UCB**

Logical Sector	Mini Sector	Offset Address <sup>1)</sup>	Size
UCB0	DFLASH0_UCB_MS0	40'0000 <sub>H</sub>	512 byte
UCB1		40'0200 <sub>H</sub>	512 byte
UCB2		40'0400 <sub>H</sub>	512 byte
UCB3		40'0600 <sub>H</sub>	512 byte
UCB4		40'0800 <sub>H</sub>	512 byte
UCB5		40'0A00 <sub>H</sub>	512 byte
UCB6		40'0C00 <sub>H</sub>	512 byte
UCB7		40'0E00 <sub>H</sub>	512 byte

## Non Volatile Memory (NVM) Subsystem

**Table 137 Sector Structure of DFLASH0\_UCB (cont'd)**

Logical Sector	Mini Sector	Offset Address <sup>1)</sup>	Size
UCB8	DFLASH0_UCB_MS1	40'1000 <sub>H</sub>	512 byte
UCB9		40'1200 <sub>H</sub>	512 byte
UCB10		40'1400 <sub>H</sub>	512 byte
UCB11		40'1600 <sub>H</sub>	512 byte
UCB12		40'1800 <sub>H</sub>	512 byte
UCB13		40'1A00 <sub>H</sub>	512 byte
UCB14		40'1C00 <sub>H</sub>	512 byte
UCB15		40'1E00 <sub>H</sub>	512 byte
UCB16	DFLASH0_UCB_MS2	40'2000 <sub>H</sub>	512 byte
UCB17		40'2200 <sub>H</sub>	512 byte
UCB18		40'2400 <sub>H</sub>	512 byte
UCB19		40'2600 <sub>H</sub>	512 byte
UCB20		40'2800 <sub>H</sub>	512 byte
UCB21		40'2A00 <sub>H</sub>	512 byte
UCB22		40'2C00 <sub>H</sub>	512 byte
UCB23		40'2E00 <sub>H</sub>	512 byte
UCB24	DFLASH0_UCB_MS3	40'3000 <sub>H</sub>	512 byte
UCB25		40'3200 <sub>H</sub>	512 byte
UCB26		40'3400 <sub>H</sub>	512 byte
UCB27		40'3600 <sub>H</sub>	512 byte
UCB28		40'3800 <sub>H</sub>	512 byte
UCB29		40'3A00 <sub>H</sub>	512 byte
UCB30		40'3C00 <sub>H</sub>	512 byte
UCB31		40'3E00 <sub>H</sub>	512 byte
UCB32	DFLASH0_UCB_MS4	40'4000 <sub>H</sub>	512 byte
UCB33		40'4200 <sub>H</sub>	512 byte
UCB34		40'4400 <sub>H</sub>	512 byte
UCB35		40'4600 <sub>H</sub>	512 byte
UCB36		40'4800 <sub>H</sub>	512 byte
UCB37		40'4A00 <sub>H</sub>	512 byte
UCB38		40'4C00 <sub>H</sub>	512 byte
UCB39		40'4E00 <sub>H</sub>	512 byte

## Non Volatile Memory (NVM) Subsystem

**Table 137 Sector Structure of DFLASH0\_UCB (cont'd)**

Logical Sector	Mini Sector	Offset Address <sup>1)</sup>	Size
UCB40	DFLASH0_UCB_MS5	40'5000 <sub>H</sub>	512 byte
UCB41		40'5200 <sub>H</sub>	512 byte
UCB42		40'5400 <sub>H</sub>	512 byte
UCB43		40'5600 <sub>H</sub>	512 byte
UCB44		40'5800 <sub>H</sub>	512 byte
UCB45		40'5A00 <sub>H</sub>	512 byte
UCB46		40'5C00 <sub>H</sub>	512 byte
UCB47		40'5E00 <sub>H</sub>	512 byte

1) System Address = DFLASH0 Base Address (see Memory Map) + Offset Address

**Attention:** A single UCB logical sector may be erased with one erase command triggered via Host Command Interpreter. Separate erase commands are required to erase multiple UCB logical sectors.

**Table 138 Sector Structure of DFLASH0\_CFS**

Logical Sector	Mini Sector	Offset Address <sup>1)</sup>	Size
CFS0	DFLASH0_CFS_MS0	80'0000 <sub>H</sub>	4 Kbyte
CFS1	DFLASH0_CFS_MS1	80'1000 <sub>H</sub>	4 Kbyte
CFS2	DFLASH0_CFS_MS2	80'2000 <sub>H</sub>	4 Kbyte
CFS3	DFLASH0_CFS_MS3	80'3000 <sub>H</sub>	4 Kbyte
CFS4	DFLASH0_CFS_MS4	80'4000 <sub>H</sub>	4 Kbyte
CFS5	DFLASH0_CFS_MS5	80'5000 <sub>H</sub>	4 Kbyte
CFS6	DFLASH0_CFS_MS6	80'6000 <sub>H</sub>	4 Kbyte
CFS7	DFLASH0_CFS_MS7	80'7000 <sub>H</sub>	4 Kbyte
CFS8	DFLASH0_CFS_MS8	80'8000 <sub>H</sub>	4 Kbyte
CFS9	DFLASH0_CFS_MS9	80'9000 <sub>H</sub>	4 Kbyte
CFS10	DFLASH0_CFS_MS10	80'A000 <sub>H</sub>	4 Kbyte
CFS11	DFLASH0_CFS_MS11	80'B000 <sub>H</sub>	4 Kbyte
CFS12	DFLASH0_CFS_MS12	80'C000 <sub>H</sub>	4 Kbyte
CFS13	DFLASH0_CFS_MS13	80'D000 <sub>H</sub>	4 Kbyte
CFS14	DFLASH0_CFS_MS14	80'E000 <sub>H</sub>	4 Kbyte
CFS15	DFLASH0_CFS_MS15	80'F000 <sub>H</sub>	4 Kbyte

1) System Address = DFLASH0 Base Address (see Memory Map) + Offset Address

### 6.2.3.4 Data Flash Bank DFLASH1

DFLASH1 includes DFLASH1\_EEPROM (single ended mode at [Table 139](#) and complement sensing mode at [Table 140](#)).

## Non Volatile Memory (NVM) Subsystem

### DFLASH1\_EEPROM Erase Operations and Erase Disturb

The number of erase operations over lifetime is limited by the parameter  $N_{ERD1}$  ( $N_{ERD1S}$  for single ended and  $N_{ERD1C}$  for complement sensing). All sectors must be used round-robin in order to prevent the accumulation of erase disturbs in static sectors<sup>1)</sup>. In derivatives with a reduced amount of DFLASH all sectors in this reduced DFLASH range must be used round-robin.

**Table 139 Sector Structure of DFLASH Bank 1 in Single Ended Mode**

Logical Sector	Physical Sector	Offset Address	Size	Total
EEPROM0	DFLASH1_EEPROM	00'0000 <sub>H</sub>	4 Kbyte	4 Kbyte
EEPROM1		00'1000 <sub>H</sub>	4 Kbyte	8 Kbyte
EEPROM2		00'2000 <sub>H</sub>	4 Kbyte	12 Kbyte
EEPROM3		00'3000 <sub>H</sub>	4 Kbyte	16 Kbyte
...		...	...	...
EEPROM28		01'C000 <sub>H</sub>	4 Kbyte	116 Kbyte
EEPROM29		01'D000 <sub>H</sub>	4 Kbyte	120 Kbyte
EEPROM30		01'E000 <sub>H</sub>	4 Kbyte	124 Kbyte
EEPROM31		01'F000 <sub>H</sub>	4 Kbyte	128 Kbyte

**Attention:** For DFLASH1\_EEPROM configured in single ended mode the minimum erase size is 4 Kbyte aligned to the logical sector address boundary.

**Table 140 Sector Structure of DFLASH Bank 1 in Complement Sensing Mode**

Logical Sector	Physical Sector	Offset Address	Size	Total
EEPROM0	DFLASH1_EEPROM	00'0000 <sub>H</sub>	2 Kbyte	2 Kbyte
EEPROM1		00'0800 <sub>H</sub>	2 Kbyte	4 Kbyte
EEPROM2		00'1000 <sub>H</sub>	2 Kbyte	6 Kbyte
EEPROM3		00'1800 <sub>H</sub>	2 Kbyte	8 Kbyte
...		...	...	...
EEPROM28		00'E000 <sub>H</sub>	2 Kbyte	58 Kbyte
EEPROM29		00'E800 <sub>H</sub>	2 Kbyte	60 Kbyte
EEPROM30		00'F000 <sub>H</sub>	2 Kbyte	62 Kbyte
EEPROM31		00'F800 <sub>H</sub>	2 Kbyte	64 Kbyte

**Attention:** For DFLASH1\_EEPROM configured in complement sensing mode the minimum erase size is 2 Kbyte aligned to the logical sector address boundary.

### 6.2.4 Program Flash (PFLASH) Features

- PFLASH Non Volatile Memory (NVM)
- Low latency safe instruction fetch path to support high performance:

1) The parameter  $N_{DFD}$  (see Data Sheet) documents the erase disturb limit. It is chosen higher than needed for pure round-robin usage to cover also error cases (e.g. repetition of erase commands on the same range due to power failures during operation).

## Non Volatile Memory (NVM) Subsystem

- DPI point-to-point interface between local CPU and local PFLASH.
- Flash prefetch buffers for linear speculative code fetches.
- Flash read access:
  - Read margins to check for sub optimal logic 0's and 1's.
- Command interface to support the following operations:
  - NVM program and erase operations.
  - Fast programming of 32 byte pages for PFLASH
  - High throughput burst programming of 256 byte units for PFLASH
  - High throughput erase by multi-sector erase commands.
  - Suspend and resume operations to interrupt on-going NVM operation.
  - Fast suspend to read command.
  - End of erase and program operations reported by interrupt.
- Password based read and write protection.
- Erase counters.
- Interrupt service requests to signal:
  - Indication of end of busy
  - Indication of an operation error
  - Indication of a protection error
  - Indication of a command sequence error
  - Indication of a program or erase verify error
- ECC Detection and Correction
  - DECTED: single-bit and double-bit error correction and triple-bit error detection.
  - Error reporting to the Safety Management Unit (SMU).
- Low power Standby Mode.
- Delivery in erased state.

### 6.2.5 Data Flash (DFLASH) Features

- DFLASH Bank 0 (DFLASH0) NVM instances
  - Multiple EEPROMx sectors commonly used for EEPROM emulation for run time application data storage.
  - User Configuration Blocks (UCBs) for protection data.
  - Configuration Sector (CFS) is a logical sector not directly accessible by the user.
- DFLASH Bank 1 (DFLASH1) NVM instances
  - Multiple EEPROMx sectors
  - In devices with HSM, they are used by the HSM for EEPROM emulation protected from application access.
- Flash read access:
  - Unique 64-bit read access to DFLASH arrays.
  - Read margins to check for sub optimal logic 0's and 1's.
- Command interface to support the following operations:
  - NVM program and erase operations.
  - Fast programming of 8 byte pages for DFLASH

## Non Volatile Memory (NVM) Subsystem

- High throughput burst programming of 32 byte units for DFLASH
- High throughput erase by multi-sector erase commands.
- Suspend and resume operations to interrupt on-going NVM operation.
- Fast suspend to read command.
- End of erase and program operations reported by interrupt.
- Password based read and write protection.
- Interrupt service requests to signal:
  - Indication of end of busy
  - Indication of an operation error
  - Indication of a protection error (DFLASH0 only)
  - Indication of a command sequence error
  - Indication of a program or erase verify error
- ECC Detection and Correction
  - TECQED: dynamic correction of single-bit, double-bit and triple-bit errors and detection of quad-bit errors.
- Pad supply voltage used for program and erase.
- Low power Standby Mode.
- Delivery of all user area in erased state.

### 6.2.6 Boot ROM (BROM) Features

- Startup SoftWare (SSW) executed at every reset.
- Test firmware to support IFX product test routines.
- Tuning Protection (TP) code to implement the “Secure Watchdog”.

## Non Volatile Memory (NVM) Subsystem

### 6.3 Safety Measures

The Flash Interface modules include the following functional safety features:

- Safety ENDINIT protection of relevant DMU configuration registers and Command Sequences modifying PFLASH content.
- Master specific access control of DMU control and configuration registers.
- Master specific access control of PFLASH read.
- Protection of safety relevant configuration registers against Single Event Effects (SEE).
- Integrity of read data stored in the NVM is ensured by an ECC checksum.
- Integrity of PFLASH read path is ensured by the following means: (For more details on the below safety mechanisms, please refer PFI chapter)
  - PFI partial lockstep mechanism
  - Protection of PFLASH wait cycles with ECC checksum
  - Protection of read data from PFI to CPU by ECC checksum
  - Additional safety mechanism to check that there is no NVM operation in the local PFLASH when it is not expected by PFI/DMU.
  - Monitoring of read parameters in the FSI (MISR, redundant Flip Flops)

#### 6.3.1 Safety Endinit protection

All command sequences that can modify PFLASH content are protected by Safety Endinit. A PROER is generated if there is an access to the PFLASH without removal of Safety Endinit protection.

In order to support “Software update Over the Air (SOTA)”, if DMU\_HP\_PROCONTP.SWAPEN is “Enabled”, then DMU removes Safety Endinit protection of the inactive PFLASHes, i.e., those PFLASHES that are not currently used for code execution by the running application. DMU uses the SCU\_SWAPCTRL bits to determine whether the system is in the standard or alternate address map, and from it, derives which are the ‘active’ and ‘inactive’ PFLASH banks. The alternate address map is described in the MEMMAP chapter. The ‘active’ banks in the standard or alternate address map mode are the PFLASH banks with the physical address (as described in the standard address map) of PFLASH0, PFLASH1 and PFLASH 4 for the AURIXTC39x and AURIXTC38x derivatives. The remaining banks are the ‘inactive’ banks. For the other derivatives, the ‘active’ bank is the one with the physical address of PFLASH0.

If an illegal value of SCU\_SWAPCTRL is detected, DMU generates a SQER when the next program/erase of any PFLASH bank is requested.

#### 6.3.2 Access Control

The Flash Module Registers located in the DMU are accessed through the SRI DFLASH slave interface. The DMU register set includes access protection registers and NVM configuration, control and status registers. The register access modes are described in the Introduction chapter.

**Table 141 Functional Safety Features of registers and PFLASH Bank**

	PFLASH Bank	Registers
Master Specific Access Control in DMU <sup>1)</sup>	–	Restricts Write Accesses
Master Specific Access Control in CPU	Restricts reads	-
ENDINIT	–	Restricts Write Accesses

## Non Volatile Memory (NVM) Subsystem

**Table 141 Functional Safety Features of registers and PFLASH Bank (cont'd)**

	PFLASH Bank	Registers
Safety ENDINIT	Restricts Command Sequences	Restricts Write Accesses
Supervisor Mode	-	Restricts Write Accesses

- 1) The master specific access control is configured and controlled by the registers DMU\_HF\_ACCENO and DMU\_HF\_ACCEN1. The ACCEN registers determine which master is allowed to perform writes to the protected DMU registers. The SRI slave interface is protected by this mechanism. Write accesses that are blocked by this mechanism create a bus error as response.

**Note:** *It is convention to use short register names (e.g. "ID") in the chapter that defines these registers. In all other chapters and in the development tools long register names are used that are a concatenation of the module instance (e.g. "DMU"), an underscore and the short register name, i.e. "DMU\_HF\_ID". This document uses for clarification also mostly the short register names.*

### 6.3.3 Data Reliability and Integrity

The data is stored in Flash with error correcting codes "ECC" in order to protect against data corruption. The reliability of Flash data can be checked with margin checks.

The following measures ensure the integrity for reading from Flash is a single point failure metric of >99% and a latent fault metric of >90%.

#### 6.3.3.1 PFLASH ECC

The ECC checksum is calculated over 256 data bits and the address bits. The ECC has the following features:

- Correction of 1-bit and 2-bit errors.
- Detection of 100% of 1-bit, 2-bit and 3-bit errors.
- Detection of >99% of all error vectors in the white noise error model.
- Detection of >99% of all-0 and all-1 cases.
- Detection of addressing errors.

As side effect of the all-0 error detection an erased Flash range can't be read without ECC errors. Also over-programming of Flash ranges with all-1 would create entries with ECC errors.

The ECC is automatically generated when programming the Flash when this is not disabled with bit field DMU\_HF\_ECCW.PECENCDIS.

The ECC is automatically evaluated when reading data. Errors are only reported for 256-bit data blocks for which at least one byte is read by the CPU port.

An errored internal prefetch data is discarded and is not stored in the prefetch buffer. The reported error is not recorded in the status registers, and will not trigger a SMU alarm.

The flash interface does not know if the master uses the requested data or discards it (e.g. due to speculatively fetching code). Therefore speculatively fetched data gets the same error response.

Error reporting and ECC disabling:

- Single-bit error:
  - Is noted in Flash bit field PFIz\_ECCS.ERR1.
  - Single Bit Error (SBER) alarm is generated.
  - If CPUx\_FLASHCON2.RECDIS = 10<sub>B</sub> then the affected address is stored uniquely in the SBAB and safety alarm is generated on SBAB full. No alarm is generated if CPUx\_FLASHCON2.RECDIS = 01<sub>B</sub>.
  - Data and ECC value are corrected if this is not disabled with Flash CPUx\_FLASHCON2.ECCCORDIS<sup>1)</sup>.

## Non Volatile Memory (NVM) Subsystem

- Double-bit error:
  - Is noted in Flash bit field PFIz\_ECCS.ERR2.
  - Double Bit Error (DBER) alarm is generated.
  - If CPUx\_FLASHCON2.RECDIS = 10<sub>B</sub> then the affected address is stored uniquely in the DBAB and safety alarm generated on DBAB full.
  - Data and ECC value are corrected if this is not disabled with Flash bit field CPUx\_FLASHCON2.ECCCORDIS.
- Multi-bit error and not All-0 error:
  - Is noted in Flash bit field PFIz\_ECCS.ERRM by the MULTIFAIL status.
  - Causes a bus error.
  - If CPUx\_FLASHCON2.RECDIS = 10<sub>B</sub> then the affected address is stored uniquely in the MBAB and safety alarm is generated.
  - An address error will be reported as multi-bit error when found.
- Multi-bit error and All-0 error:
  - Is noted in Flash bit field PFIz\_ECCS.ERRM by the MULTIFAIL status.
  - Causes a bus error.
  - If CPUx\_FLASHCON2.RECDIS = 10<sub>B</sub> then the affected address is stored uniquely in MBAB and ZBAB and safety alarm is generated. Safety alarm is also generated on ZBAB full.
  - An address error will be reported as multi-bit error when found.

### 6.3.3.2 DFLASH ECC

The ECC is calculated only over 64 data bits. Therefore addressing faults (correct data is read from an incorrect address) can not be detected.

The algorithm has the following advantages:

- An erased Flash range delivers ECC correct 0 data in the single ended sensing mode (default option). If the complement sensing mode is selected, then an erased section contains cell pairs (erased/erased) which are not compliant to complement data structure (prog/erased or erased/prog) and may result in an arbitrary read data result together with any possible ECC correction (e.g. wrong correction or multibit error).
- Over-programming with all-1 delivers an ECC correct result for both sensing modes. However, in complement sensing mode, the all-1 read data may not have the full retention specified in the datasheet because in most cases, it would not be well defined complement data pattern (prog/prog instead of prog/erase).

### 6.3.4 Integrity of PFlash read data wait cycles

Wait cycle configured in DMU\_HF\_PWAIT register is protected by ECC check sum. This ECC is transported along with wait cycle information from DMU to each of the PFIs. The PFI performs an ECC check on the received data every cycle and reports an error to CPU (resulting in PFLASH read path monitor alarm to SMU). This ECC provides detection of 1-bit and 2-bit errors.

### 6.3.5 Alarms

The Flash Interface modules provide the following alarms to the SMU:

- PFRWB corrected Single Bit error Address Buffer (SBAB) full: OR'ed for all PFRWB instances.
- PFRWB corrected Double Bit error Address Buffer (DBAB) full: OR'ed for all PFRWB instances.

1) Disabling the correction of errors with CPUx\_FLASHCON2.ECCCORDIS is a test feature. During application run-time the correction must be enabled.

## Non Volatile Memory (NVM) Subsystem

- PFRWB uncorrected Multi Bit error Address Buffer (MBAB) full: OR'ed for all PFRWB instances.
- PFRWB uncorrected all Zeros Bits error Address Buffer (ZBAB) full: OR'ed for all PFRWB instances.
- PFRWB Single Bit Error (SBER) : OR'ed for all PFRWB instances.
- PFRWB Double Bit Error (DBER) : OR'ed for all PFRWB instances.
- NVM Configuration Error (NVMCERR) : OR'ed for all FSI and PFRWB instances.
- PFRWB EDC error detected in the EDC checker: OR'ed for all PFRWB instances.
- PFRWB ECC error detected in on-line ECC checker : OR'ed for all PFRWB instances.
- PFRWB FLASHCON error (FLCONERR) - Illegal FLASHCON register values detected by PFRWB : OR'ed for all PFRWB instances
- PFLASH read path monitor error : All errors in PFI are OR'ed and sent via its local CPU to SMU.
- DMU SRI slave interface address phase error (see [Chapter 6.3.5.1](#)).
- DMU SRI slave interface write data phase error (see [Chapter 6.3.5.2](#)).

### 6.3.5.1 SRI Access Address Phase Error

If an ECC error occurs during the address phase of an SRI access then the DMU\_HF\_ERRSR.ADER bit will be set and an error will be signalled to the SMU. The SRI access will terminate with an error.

### 6.3.5.2 SRI Access Write Data Phase Error

If an ECC error occurs on the data phase of an SRI write access then an error will be signalled to the SMU.

## Non Volatile Memory (NVM) Subsystem

### 6.4 Revision History

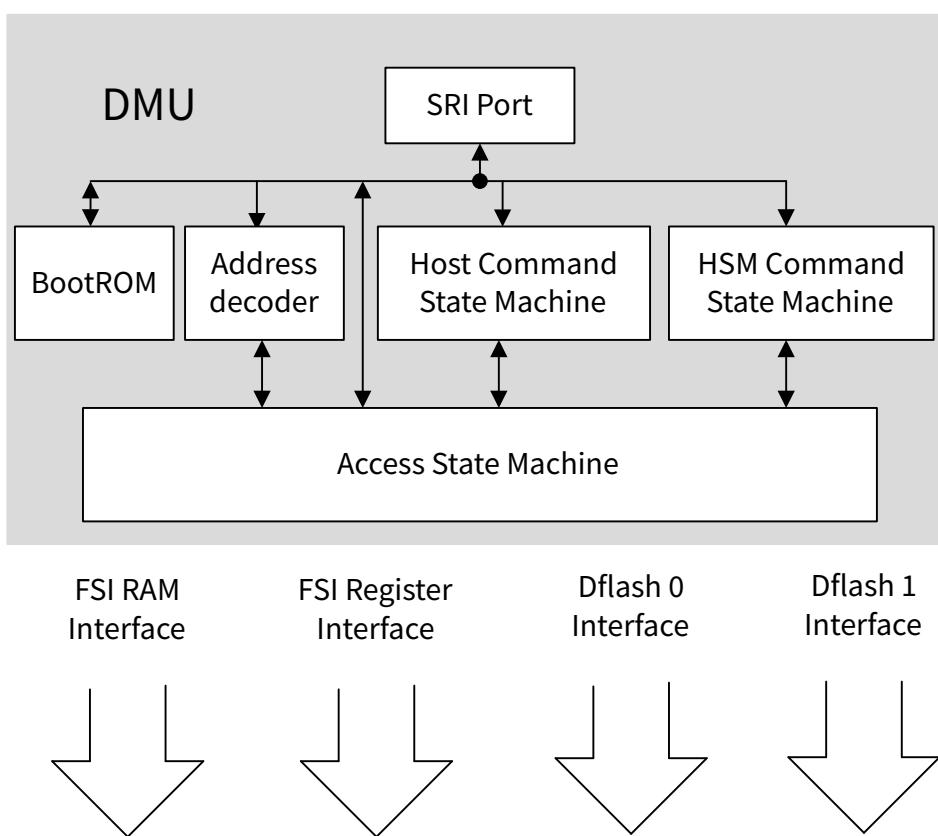
**Table 142 Revision History**

Reference	Change to Previous Version	Comment
<b>V2.0.3</b>		
<a href="#">Chapter 6.1</a>	Corrected read protection in security layer overview to be global rather than bank granularity.	
<b>V2.0.4</b>		
<a href="#">Page 17</a>	<a href="#">Chapter 6.3.3.1</a> - Alarm generated on xBAB full rather than overflow.	
<a href="#">Page 18</a>	<a href="#">Chapter 6.3.3.1</a> - Clarified address error/multi-bit error reporting.	
<a href="#">Page 18</a>	<a href="#">Chapter 6.3.5</a> - Alarm generated on xBAB full rather than overflow.	
<b>V2.0.5</b>		
<a href="#">Page 5</a>	Changed register name from DMU_SF_PROCONHSMCX0-1 to DMU_SP_PROCONHSMCX0-1 and DMU_SF_PROCONHSMCOTP0-1 to DMU_SP_PROCONHSMCOTP0-1 in <a href="#">Chapter 6.2.3.1.1</a> .	

## 6.5 Data Memory Unit (DMU)

### 6.5.1 Overview

The DMU (in [Figure 59](#)) facilitates reads of DFLASH0, DFLASH1, UCB and CFS and facilitates all operations executed on the PFLASH and DFLASH memories. DMU also contains the BootROM, all reads of which are through the SRI interface. DMU interfaces to the FSI and PFI for all flash operations, and PFlash read respectively.



**Figure 59 Block diagram of DMU module**

The DMU supports the following features:

- Support of [Command Sequences](#) to perform flash operations to all PFLASH and DFLASH banks and UCBs.
- Facilitates DFLASH0, DFLASH1, UCB and CFS reads
- Functions as the security layer over the flash - controls access to all the flash banks based on the protection installed in the UCBs. Performs password based protection of the UCBs and Flash banks. Has two separate channels for Host and HSM programming of DFLASH1
- Performs HSM exclusivity checks and controls access to DFLASH1 based on the HSM configuration programmed in the UCBs

## 6.5.2 Functional Description

### 6.5.2.1 Flash Read Access

If the Flash banks are in Read Mode then all flash read accesses are memory mapped reads. Read protection can be used to block a read.

#### 6.5.2.1.1 Transaction Types

##### Program Flash

The transaction types for local CPU (at [Table 143](#)) and remote CPU (at [Table 144](#)) read accesses to a PFLASH are as follows:

**Table 143 Local CPU Read Accesses to PFLASH**

Local CPU access	Bus	Code Fetch	Data Constant
Cacheable address	DPI	Block Transfer 4 (BTR4)	Block Transfer 4 (BTR4)
Non-cacheable address	DPI	Block Transfer 4 (BTR4)	Minimum width

**Table 144 Remote CPU Read Accesses to PFLASH**

CPU access	Bus	Code Fetch	Data Constant
Cacheable address	SRI	Block Transfer 4 (BTR4)	Block Transfer 4 (BTR4)
Non-cacheable address	SRI	Block Transfer 4 (BTR4)	Minimum width

##### Data Flash

Read accesses to DFLASH must be single transfers made across the SRI and are available only in the non-cacheable address range. A Block Transfer will result in a bus error.

#### 6.5.2.1.2 Configuring Flash Read Access Cycles

The Flash read path including the ECC decoders uses the  $f_{SRI}$  clock. The minimum number of cycles can be calculated based on the delays given in the Data Sheet or the Electrical Specification.

**Attention:** In case of using the FM-PLL the maximum frequency of the  $f_{SRI}$  clocks has to enter the calculation.

##### Configuring Program Flash Read Access Cycles

Read accesses to the PFLASH have two delay settings counted with the  $f_{SRI}$  clock. The settings are defined in the [HF\\_PWAIT](#) register for:

- [Operation Mode](#) including [Error Mode](#)
- [Cranking Mode](#)

**Table 145 PFLASH Read Access Cycle Calculation**

	<b>Minimum Value<sup>1)</sup></b>
PFLASH read cycles	Ceiling( $t_{PF} * f_{SRI}$ )
ECC cycles	Ceiling( $t_{PF ECC} * f_{SRI}$ )

1) The Ceiling(r) function rounds up a real number, i.e., the result is the smallest integer not less than the real argument.

**Attention:** Note that the wait cycles to be programmed in the DMU\_HF\_PWAIT register is ‘PFLASH read cycles’ - 1, and ‘ECC cycles’ - 1 as shown in the example below.

### Program Flash Example

Example configuration for  $t_{PF} = 30$  ns and  $t_{PF ECC} = 10$  ns, with  $f_{SRI} = 200$  MHz in **Operation Mode**. The number of PFLASH read cycles equals 6 and the number of error correction cycles equals 2 therefore the wait cycle values to be programmed are:

- **HF\_PWAIT.RFLASH** = 5<sub>D</sub>
- **HF\_PWAIT.RECC** = 1<sub>D</sub>

### Configuring Data Flash Read Access Cycles

Read accesses to the DFLASH have one delay setting counted with the  $f_{FSI}$  clock. The settings are defined in the **HF\_DWAIT** register.

**Table 146 DFLASH Read Access Cycle Calculation**

	<b>Minimum Value<sup>1)</sup></b>
DFLASH read cycles	Ceiling( $t_{DF} * f_{FSI}$ )
ECC cycles	Ceiling( $t_{DF ECC} * f_{FSI}$ )

1) The Ceiling(r) function rounds up a real number, i.e., the result is the smallest integer not less than the real argument.

**Attention:** Note that the wait cycles to be programmed in the DMU\_HF\_DWAIT register is ‘DFLASH read cycles’ - 1, and ‘ECC cycles’ - 1 as shown in the example below.

### Data Flash Example

Example configuration for  $t_{DF} = 100$  ns and  $t_{DF ECC} = 20$  ns, with  $f_{FSI} = 100$  MHz. The number of DFLASH read cycles equals 10 and the number of error correction cycles equals 2 therefore the wait cycle values to be programmed are:

- **HF\_DWAIT.RFLASH** = 9<sub>D</sub>
- **HF\_DWAIT.ECC** = 1<sub>D</sub>

### 6.5.2.2 Flash Operations

All Flash operations except memory mapped reads are performed with command sequences. Write accesses are treated as follows:

- Write accesses to the PFLASHp memory range are refused with bus error.
- Write accesses to the DFLASH0 and DFLASH1 memory ranges are interpreted as a command cycles belonging to the respective command sequences.

The DMU has a Command Sequence Interpreter (CSI) to process command sequences.

In devices with HSM, the DMU supports two Command Sequence Interpreters (CSIs):

- Host Command Sequence Interpreter
  - The DMU executes command sequences on all PFLASH banks, DFLASH0 and DFLASH1 when DFLASH1 is not HSM exclusive.
  - The Host command sequence interpreter decodes write accesses to the DFLASH0 address range in the memory map segment A.
  - Any on chip bus master can perform command sequences via the Host Command Sequence Interpreter.
- HSM Command Sequence Interpreter
  - The DMU executes command sequences on DFLASH1 when DFLASH1 is HSM exclusive.
  - The HSM command sequence interpreter decodes write accesses to the DFLASH1 address ranges in the memory map segments A and F.
  - Only on chip bus masters marked with the Access Term Symbol H can perform command sequences via the HSM Command Sequence Interpreter.

### 6.5.2.2.1 Page Mode

A Command Sequence Interpreter must be in **Page Mode** to:

- Load write data into the Assembly Buffer (ASB).
- Initiate a write command to program data into PLFLASH or DFLASH.

### 6.5.2.2.2 Command Sequences

Command sequences consist of 1 to 9 command cycles. The command sequence interpreter checks that a command cycle is correct in the current state of command interpretation else a **Sequence Error (SQER)** is reported.

The DMU includes a Host Command Interface (see [Chapter](#)) and a HSM Command Interface (see [Chapter](#)). Both interfaces are completely independent. They can issue command sequences independently to the FSI and they see only the results, errors and busy signals of their own commands<sup>1)</sup>. The FSI arbitrates between the commands from both interfaces. Common resources in FSI and Flash are shared in time slices.

Command sequences to PFLASH are blocked by the Safety Endinit protection. See ‘Functional Safety Features’ section in NVM Subsystem chapter for more details on this protection.

Register read and write accesses are not affected by active commands.

#### Host Command Sequence Interpreter

The Host command sequence interpreter:

- Interprets writes to the DFLASH0 EEPROM memory range as command cycles.
- Reports status and errors in **HF\_STATUS** and **HF\_ERRSR** registers.
- Uses its own assembly buffer for programming data.
- Can be used to perform operations on PFLASH, DFLASH0, UCBs and DFLASH1<sup>2)</sup>.

Triggering an NVM operation using the command sequences takes the DMU into Command Mode. During its execution the Flash bank reports BUSY in DMU\_HF\_STATUS. In this mode read accesses to a Flash bank are refused with a bus error or the ready is suppressed until BUSY clears. The read access response is configured as follows:

- PFLASH response is configured on a bank granularity by CPUX\_FLASHCON1.STALL

1) If however the DFLASH1 is busy with a command received from the Host command interface (possible as long DFLASH1 is not HSM\_exclusive) commands from the HSM command interface are refused with a bus error.

2) DFLASH1 operations are controlled by “HSMDX” and “BLKFLAN”.

At the end of a flash operation the Flash bank clears the BUSY bit and read accesses are enabled. Only flash operations with a significant duration (shown in the command documentation) set BUSY.

**Attention:** *Using CPUx\_FLASHCON1.STALL = “STALL” is not recommended because the stalled CPU is inoperable. This feature should be only used under controlled conditions by a Flash loader.*

**Attention:** *Using CPUx\_FLASHCON1.STALL = “STALL” together with Flash sleep is prohibited as it might lead to device hang-up.*

If a command is executing then further command sequences to this command sequence interpreter are not allowed. Any writes to the command sequence interpreter are refused with a bus error. Generally when the command sequence interpreter detects an error it reports a sequence error by setting **HF\_ERRSR.SQER** or a protection error by setting **HF\_ERRSR.PROER**. Then the command sequence interpreter is reset and a **Page Mode** is left. The next command cycle must be the 1st cycle of a command sequence. The only exception is “Enter Page Mode” when a Flash is already in **Page Mode**.

### HSM Command Sequence Interpreter

The HSM command sequence interpreter:

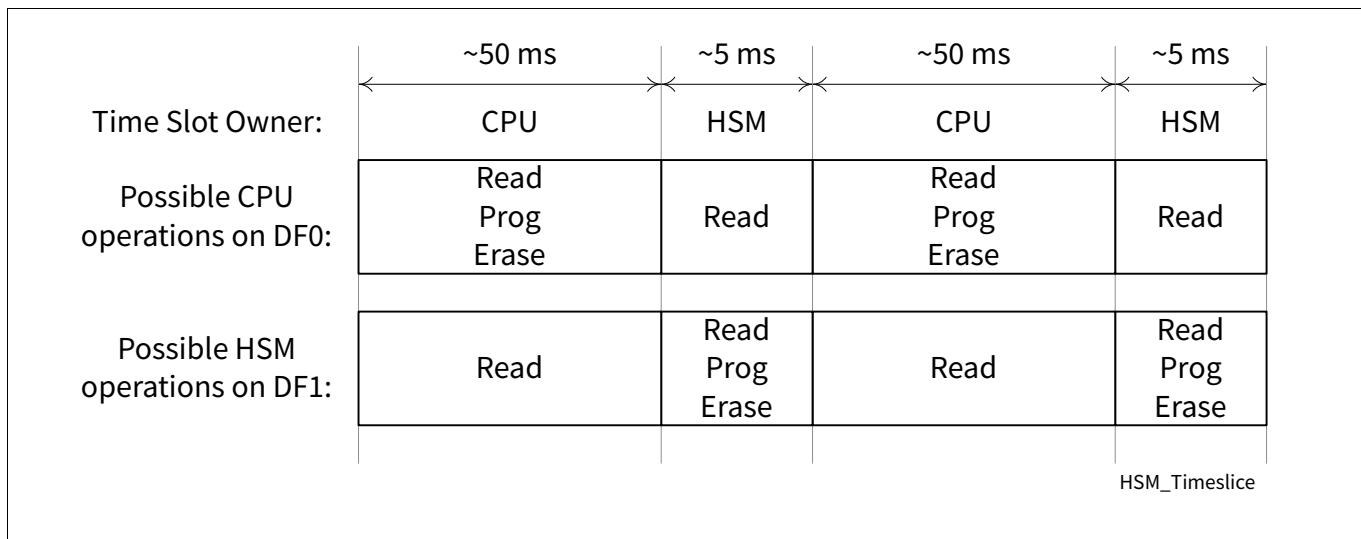
- Interprets writes to the DFLASH1 EEPROM memory range as command cycles.
- Writes by any other master than HSM (and Cerberus when HSM debugging is enabled) are refused with a bus error.
- Reports status and errors in **SF\_STATUS** and **SF\_ERRSR** registers.
- Uses its own assembly buffer for programming data.
- Can be used to perform operations on DFLASH1.

The HSM has a dedicated command interface. If a command is executing then further command sequences to this command sequence interpreter are not allowed. Any writes to the command sequence interpreter are refused with a bus error.

### Time Slice Control

The FSI performs Flash commands by the CPU and HSM in time slices (see [Figure 60](#)):

- CPU time slice of 50 ms.
- HSM time slice of 5 ms.



**Figure 60 Time Slice Control Overview**

Time slicing starts automatically when two requests (one HSM and one CPU) compete for resources. The first request starts execution immediately. The second request then triggers the time slice handling. During the CPU time slice phase the command from Host command sequence interpreter is executed and in HSM time slice phase the command from HSM. Switching between the time slice phases is done by firmware. Busy is high also for the inactive phase in time slice, so the functionality is not visible from outside.

In the HSM time slice an HSM command is executed. The roles of HSM and CPU are interchanged.

The following examples describe how conflicts are handled. The examples show a DFLASH0 access on Host CSI and it is to be noted that the same examples are valid for any PFLASH accesses on Host CSI.

DFLASH0 busy with erase:

- DFLASH0 can be only read after suspending the erase with **HF\_SUSPEND.REQ**
- Also for programming in DFLASH0 the erase has to be suspended or finished.
- The DFLASH1 can be read by HSM.
- A program/erase job in DFLASH1 is performed for 5 ms after the CPU time slice of 50 ms has expired.

DFLASH1 busy with HSM erase:

- DFLASH1 can be only read after suspending the erase with **SF\_SUSPEND.REQ**
- Also for programming in DFLASH1 the erase has to be suspended or finished.
- The DFLASH0 can be read by the CPU.
- A program/erase job in DFLASH0 is performed for 50 ms after the HSM time slice of 5 ms has expired.

DFLASH0 busy with programming:

- DFLASH0 can be only read after the programming has finished or has been suspended.
- Also for erasing in DFLASH0 the programming job has to finish or has to be suspended.
- A DFLASH0 programming job received during HSM time slice is started after the HSM time slice expired.
- The DFLASH1 can be read by HSM.
- A program/erase job in DFLASH1 is performed for 5 ms after the CPU time slice of 50 ms has expired.

DFLASH1 busy with HSM programming:

- DFLASH1 can be only read after the programming has finished or has been suspended.
- Also for erasing in DFLASH1 the programming job has to finish or has to be suspended.
- A DFLASH1 programming job received during the CPU time slice is started after the CPU time slice expired.

- The DFLASH0 can be read by the CPU.
- A program/erase job in DFLASH0 is performed for 50 ms after the HSM time slice of 5 ms has expired.

### Time Slice Control and Flash Parameters

The duration of program and erase commands is described in the data sheet.

The DFLASH erase times ( $t_{ERD}$ ,  $t_{MERD}$ ) are documented for the case that only one operation is executed uninterrupted by the time slicing. In the case of two concurrent operations (i.e. active time slicing) the duration increases, about 15% for CPU erase commands and by a factor of about 15 for HSM erase commands.

The programming durations of DFLASH operations ( $t_{PRD}$  and  $t_{PRDB}$ ) are also given for commands running unaffected by the time slice operation. The execution of programming commands issued by the CPU can be shifted by up to 5 ms and for those issued by the HSM can be shifted by up to 50 ms. From outside this appears as prolonged busy times by 5 ms or 50 ms.

#### 6.5.2.2.3 Command Sequence Definitions

The command sequence descriptions use the following nomenclature (symbolic assembly language):

**ST addr, data:** Symbolic representation of a command cycle moving “data” to “addr”.

The parameter “addr” is defined as followed:

- **CCCC<sub>H</sub>:** The “addr” must point into any of the address regions of the Command Sequence Interpreter used. The last 16 address bits must match CCCC<sub>H</sub>.

The parameter “data” can be one of the following:

- **PA:** Absolute start address of the Flash page. Must be aligned to burst size for “Write Burst” or to the page size for “Write Page”.
- **WA:** Absolute start address of the Flash wordline.
- **SA:** Absolute start address of a Flash sector. Allowed are the PFLASH sectors Sx and the DFLASH sectors EEPROMx and UCBx.
- **WD:** 64-bit or 32-bit write data to be loaded into the page assembly buffer.
- **xxYY:** 8-bit write data as part of a command cycle. Only the byte “YY” is used for command interpretation. The higher order bytes “xx” are ignored.
  - **xx5y:** Specific case for “YY”. The “y” can be “0<sub>H</sub>” for selecting the PFLASH or “D<sub>H</sub>” to select the DFLASH.
  - **xxww:** Specific case for “YY”. The “ww” defines the number of pages and must not cross a wordline border.
  - **xxnn:** Specific case for “YY”. The “nn” defines the number of logical sectors that are erased or verified. This number underlies certain restrictions (see [Erase Logical Sector Range](#)).
- **UC:** Identification of the UCBx for which the password checking shall be performed:
  - xx00<sub>H</sub> for UCB\_BMHxDn\_ORIG and UCB\_BMHxDn\_COPY (n=0 - 3)
  - xx10<sub>H</sub> for UCB16 = UCB\_PFLASH\_ORIG and UCB24 = UCB\_PFLASH\_COPY to disable global read and write protection for all PFLASHes
  - xx11<sub>H</sub> for UCB17 = UCB\_DFLASH\_ORIG and UCB25 = UCB\_DFLASH\_COPY
  - xx12<sub>H</sub> for UCB18 = UCB\_DBG\_ORIG and UCB26 = UCB\_DBG\_COPY
  - xx16<sub>H</sub> for UCB22 = UCB\_ECPRIOR\_ORIG and UCB30 = UCB\_ECPRIOR\_COPY
  - xx17<sub>H</sub> for UCB23 = UCB\_SWAP\_ORIG and UCB31 = UCB\_SWAP\_COPY
  - xx0F<sub>H</sub> for UCB15 = UCB\_RETEST
  - xx20<sub>H</sub> for UCB16 = UCB\_PFLASH\_ORIG and UCB24 = UCB\_PFLASH\_COPY to disable global read protection, and sector specific write protection for PFLASH0

- $\text{xx21}_H$  for UCB16 = UCB\_PFLASH\_ORIG and UCB24 = UCB\_PFLASH\_COPY to disable global read protection, and sector specific write protection for PFLASH1
- $\text{xx22}_H$  for UCB16 = UCB\_PFLASH\_ORIG and UCB24 = UCB\_PFLASH\_COPY to disable global read protection, and sector specific write protection for PFLASH2
- $\text{xx23}_H$  for UCB16 = UCB\_PFLASH\_ORIG and UCB24 = UCB\_PFLASH\_COPY to disable global read protection, and sector specific write protection for PFLASH3
- $\text{xx24}_H$  for UCB16 = UCB\_PFLASH\_ORIG and UCB24 = UCB\_PFLASH\_COPY to disable global read protection, and sector specific write protection for PFLASH4
- $\text{xx25}_H$  for UCB16 = UCB\_PFLASH\_ORIG and UCB24 = UCB\_PFLASH\_COPY to disable global read protection, and sector specific write protection for PFLASH5

- **PWx:** 32-bit word of a 256-bit password.

When using for command cycles 64-bit transfers the “data” is expected in the correct 32-bit word as indicated by the address “addr”.

### Command Sequence Overview Table

The following sections describe each command sequence (at [Table 147](#)) in detail.

**Table 147 Command Sequences for Flash Control**

Command Sequence		1. Cycle	2./6. Cycle	3./7. Cycle	4./8. Cycle	5./9. Cycle	6. Cycle
<b>Reset to Read</b>	Address Data	.5554 .xx <b>F0</b>					
<b>Enter Page Mode</b>	Address Data	.5554 .xx <b>5y</b>					
<b>Load Page</b>	Address Data	.55F0 <b>WD</b>					
<b>Write Page</b>	Address Data	.AA50 <b>PA</b>	.AA58 .xx <b>00</b>	.AAA8 .xx <b>A0</b>	.AAA8 .xx <b>AA</b>		
<b>Write Page Once</b>	Address Data	.AA50 <b>PA</b>	.AA58 .xx <b>00</b>	.AAA8 .xx <b>A0</b>	.AAA8 .xx <b>A8</b>		
<b>Write Burst</b>	Address Data	.AA50 <b>PA</b>	.AA58 .xx <b>00</b>	.AAA8 .xx <b>A0</b>	.AAA8 .xx <b>A6</b>		
<b>Write Burst Once</b>	Address Data	.AA50 <b>PA</b>	.AA58 .xx <b>00</b>	.AAA8 .xx <b>A0</b>	.AAA8 .xx <b>A4</b>		
<b>Replace Logical Sector</b>	Address Data	.AA50 <b>PA</b>	.AA58 .xx <b>00</b>	.AAA8 .xx <b>A0</b>	.AAA8 .xx <b>AC</b>		
<b>Verify Erased Page</b>	Address Data	.AA50 <b>PA</b>	.AA58 .xx <b>00</b>	.AAA8 .xx <b>80</b>	.AAA8 .xx <b>56</b>		
<b>Verify Erased WL</b>	Address Data	.AA50 <b>WA</b>	.AA58 .xx <b>00</b>	.AAA8 .xx <b>80</b>	.AAA8 .xx <b>58</b>		
<b>Verify Erased Logical Sector Range</b>	Address Data	.AA50 <b>SA</b>	.AA58 .xx <b>nn</b>	.AAA8 .xx <b>80</b>	.AAA8 .xx <b>5F</b>		
<b>Erase Logical Sector Range</b>	Address Data	.AA50 <b>SA</b>	.AA58 .xx <b>nn</b>	.AAA8 .xx <b>80</b>	.AAA8 .xx <b>50</b>		

**Table 147 Command Sequences for Flash Control (cont'd)**

<b>Command Sequence</b>		<b>1. Cycle</b>	<b>2./6. Cycle</b>	<b>3./7. Cycle</b>	<b>4./8. Cycle</b>	<b>5./9. Cycle</b>	<b>6. Cycle</b>
<b>Resume NVM Operation</b>	Address Data	.AA50 <b>PA/SA</b>	.AA58 .xxnn	.AAA8 .xx <b>70</b>	.AAA8 .xx <b>CC</b>		
<b>Disable Protection</b>	Address Data	.553C <b>UC</b>	.553C <b>PW0/4</b>	.553C <b>PW1/5</b>	.553C <b>PW2/6</b>	.553C <b>PW3/7</b>	
<b>Resume Protection</b>	Address Data	.5554 .xx <b>F5</b>					
<b>Clear Status</b>	Address Data	.5554 .xx <b>FA</b>					

**Reset to Read****Calling**

- ST 5554<sub>H</sub>, xxF0<sub>H</sub>

**Function**

This function resets the addressed command sequence interpreter to its initial state (i.e. the next command cycle must be the 1st cycle of a sequence). A **Page Mode** is aborted.

This command is the only one that is accepted without **Sequence Error (SQER)** when the command sequence interpreter has already received command cycles of a different sequence. Thus **Reset to Read** can cancel every command sequence before its last command cycle has been received.

A system reset will re-initialize the NVM control circuits.

**Host Command Sequence Interpreter**

Host command sequence interpreter is reset. The following flags are cleared:

- Command sequence error flag: DMU\_HF\_ERRSR.SQER
- Protection error flag: DMU\_HF\_ERRSR.PROER
- **Page Mode** flags: DMU\_HF\_STATUS.DFPAGE and DMU\_HF\_STATUS.PFPAGE

**HSM Command Sequence Interpreter**

HSM command sequence interpreter is reset. The following flags are cleared:

- Command sequence error flag: DMU\_SF\_ERRSR.SQER
- **Page Mode** flags: DMU\_SF\_STATUS.DFPAGE

**Enter Page Mode****Calling**

- ST 5554<sub>H</sub>, xx5y<sub>H</sub>

**Function**

The PFLASH or the DFLASH assembly buffer enter **Page Mode** selected by the parameter "y".

The write pointer of the page assembly buffer is set to 0, its previous content is maintained.

If a new **Enter Page Mode** command sequence is received while any Flash is already in **Page Mode** then **Sequence Error (SQER)** is set, the existing **Page Mode** is aborted, and the new **Enter Page Mode** sequence is correctly executed (i.e. in this case the command sequence interpreter is not reset).

**Page Mode** is exited when the next command sequence terminates.

### Host Command Sequence Interpreter

**Page Mode** is signalled by the flags DMU\_HF\_STATUS.PFPAGE and DMU\_HF\_STATUS.DFPAGE for PFLASH and DFLASH respectively.

### HSM Command Sequence Interpreter

**Page Mode** is signalled by the flag DMU\_SF\_STATUS.DFPAGE.

If the DMU is in **Cranking Mode** or **Error Mode** then the command fails with an **Sequence Error (SQER)**.

### Load Page

#### Calling

- ST 55F0<sub>H</sub>, WD (Note: offset 55F4<sub>H</sub> is used for the higher order 32-bit transfers).

#### Function

Loads the data “WD” into the page assembly buffer and increments the write pointer to the next position.

All WD transfers for one page must have the same width (either all 32-bit or all 64-bit). Else the transfer is refused with SQER.

See **Chapter** for information on programming 32-bit Load Page from the CPU.

If **Load Page** is called more often than allowed by the available buffer space of 256 bytes PFLASH (32 byte DFLASH) then the overflow data is discarded and **Page Mode** is not left. This overflow is reported by the following Write Page/Burst command with **Sequence Error (SQER)**.

If the DMU is in **Cranking Mode** or **Error Mode** then the command fails with an **Sequence Error (SQER)**.

### Write Page

#### Calling

- ST AA50<sub>H</sub>, PA
- ST AA58<sub>H</sub>, xx00<sub>H</sub>
- ST AAA8<sub>H</sub>, xxA0<sub>H</sub>
- ST AAA8<sub>H</sub>, xxAH

#### Function

This function starts the programming process for one page with the data transferred previously by **Load Page** commands. Upon executing the command, **Page Mode** is exited (indicated by clearing the corresponding PAGE flag) and the BUSY flag of the bank is set.

This command is refused with **Sequence Error (SQER)** when the addressed Flash is not in **Page Mode**.

SQER is also issued when PA addresses an unavailable Flash range or when PA does not point to a legal page start address.

If after **Enter Page Mode** too few data, no data or too much data was transferred to the assembly buffer with **Load Page** then **Write Page** programs the page but sets **Sequence Error (SQER)**. Missing data is programmed with the previous content of the assembly buffer.

When the page “PA” is located in a sector with active write protection or the Flash module has an active global read protection the execution fails and **Protection Error (PROER)** is set.

When the programming process incurs an error the flag PVER is set.

The same applies to the HSM command sequence interpreter which maintains its own assembly buffer. After programming to the HSMx sectors the assembly buffer is automatically erased.

If the DMU is in **Cranking Mode** or **Error Mode** then the command fails with an **Sequence Error (SQER)**.

### Write Page Once

#### Calling

- ST AA50<sub>H</sub>, PA
- ST AA58<sub>H</sub>, xx00<sub>H</sub>
- ST AAA8<sub>H</sub>, xxA0<sub>H</sub>
- ST AAA8<sub>H</sub>, xxA8<sub>H</sub>

#### Function

This function starts the programming process for one page as the normal **Write Page** does. But before programming it checks if the page is erased. If the page is not erased (allowing correctable errors) the command fails with **Erase Verify Error (EVER)**.

When the programming itself incurs an error the flag **Program Verify Error (PVER)** is set.

On sectors with “write-once” protection only **Write Page Once** or **Write Burst Once** is accepted by DMU.

### Host Command Sequence Interpreter

The command is only supported for PFLASH.

When applied to DFLASH the command fails with DMU\_HF\_ERRSR.SQER.

### HSM Command Sequence Interpreter

The command is rejected by the HSM command sequence interpreter with DMU\_SF\_ERRSR.SQER.

If the DMU is in **Cranking Mode** or **Error Mode** then the command fails with an **Sequence Error (SQER)**.

### Write Burst

#### Calling

- ST AA50<sub>H</sub>, PA
- ST AA58<sub>H</sub>, xx00<sub>H</sub>
- ST AAA8<sub>H</sub>, xxA0<sub>H</sub>

- ST AAA8<sub>H</sub>, xxA6<sub>H</sub>

## Function

This function starts the programming process for an aligned group of pages. The programming process is more efficient than for a single page, achieving a significantly higher throughput (see data sheet).

In the PFLASH 8 pages (256 bytes) are programmed and in DFLASH 4 pages (32 bytes). The pages are programmed one after the other with increasing addresses.

## Host Command Sequence Interpreter

For disabled ECC generation:

- If DMU\_HF\_ECCW.PECENCDIS = '11<sub>B</sub>' then the user must not perform a Write Burst to PFLASH.
- If DMU\_HF\_ECCW.DECENCDIS = '11<sub>B</sub>' then the user must not perform a Write Burst to DFLASH.

## HSM Command Sequence Interpreter

For disabled ECC generation:

- If DMU\_SF\_ECCW.DECENCDIS = '11<sub>B</sub>' then the user must not perform a Write Burst to DFLASH.

If the DMU is in **Cranking Mode** or **Error Mode** then the command fails with an **Sequence Error (SQER)**.

For further details see "Write Page".

## Write Burst Once

### Calling

- ST AA50<sub>H</sub>, PA
- ST AA58<sub>H</sub>, xx00<sub>H</sub>
- ST AAA8<sub>H</sub>, xxA0<sub>H</sub>
- ST AAA8<sub>H</sub>, xxA4<sub>H</sub>

## Function

This function starts the programming process for an aligned group of pages as the normal "Write Burst" does. But before programming it checks if the pages are erased. If the pages are not erased (allowing correctable errors) the command fails with EVER.

When the programming itself incurs an error the flag PVER is set.

On sectors with "write-once" protection only **Write Page Once** or **Write Burst Once** is accepted by DMU.

For disabled ECC generation:

- If DMU\_HF\_ECCW.PECENCDIS = '11<sub>B</sub>' then the user must not perform a Write Burst Once to PFLASH.

## Host Command Sequence Interpreter

The command is only supported for PFLASH.

When applied to DFLASH the command fails with DMU\_HF\_ERRSR.SQER.

## HSM Command Sequence Interpreter

The command is rejected by the HSM command sequence interpreter with DMU\_SF\_ERRSR.SQER.

If the DMU is in **Cranking Mode** or **Error Mode** then the command fails with an **Sequence Error (SQER)**.

## Replace Logical Sector

### Calling

- ST AA50<sub>H</sub>, PA
- ST AA58<sub>H</sub>, xx00<sub>H</sub>
- ST AAA8<sub>H</sub>, xxA0<sub>H</sub>
- ST AAA8<sub>H</sub>, xxAC<sub>H</sub>

### Use Case

Replace Logical Sector is a command sequence that enables the replacement of a PFLASH logical sector with hard fails with an available redundant PFLASH logical sector. It can be used if a sector fails during an erase or program in order to meet the endurance of PFLASH (see datasheet parameter N<sub>E\_P</sub>).

There are two redundant logical sectors provided for every physical sector for the purpose of replacing a failing logical sector. These are available both for production test, and for the user. A failure in an already replaced logical sector cannot be repaired, i.e., this failing redundant logical sector cannot be mapped to an available redundant logical sector. If the command is successfully executed, UCB\_REDSEC is updated with the redundancy information. User can read UCB\_REDSEC to get information on the replaced sectors (both by user and by production test) and the available redundant sectors.

### Function

The argument “PA” is used to select a page address aligned to the start of the PFLASH logical sector to be re-mapped.

### Host Command Sequence Interpreter

The sequence error flag SQER (DMU\_HF\_ERRSR.SQER) is set when the PA is not aligned to the start address of a PFLASH logical sector page address or when PA addresses an unavailable Flash range.

The protection error flag(DMU\_HF\_ERRSR.PROER) is set when there is an attempt to re-map a logical sector with enabled **PFLASH Write Protection**.

Note that both DMU\_HF\_STATUS.PxBUSY and DMU\_HF\_STATUS.D0BUSY are set during the execution of this command.

### HSM Command Sequence Interpreter

This command is rejected by the HSM Command Sequence Interpreter with DMU\_SF\_ERRSR.SQER.

**Erase Verify Error (EVER)** is generated if there is no free entry available in UCB\_REDSEC.

**Program Verify Error (PVER)** is generated if there is a programming error during the programming of UCB\_REDSEC with the redundancy information.

If the DMU is in **Cranking Mode** or **Error Mode** then the command fails with an **Sequence Error (SQER)**.

A detailed application note is available on request.

## Verify Erased Page

### Calling

- ST AA50<sub>H</sub>, PA
- ST AA58<sub>H</sub>, xx00<sub>H</sub>
- ST AAA8<sub>H</sub>, xx80<sub>H</sub>
- ST AAA8<sub>H</sub>, xx56<sub>H</sub>

### Function

This command verifies if one page addressed by “PA” is correctly erased, i.e. contain 0 data and ECC bits. Upon executing the command, the BUSY flag of the corresponding bank is set.

As the PFLASH has Safety ECC, the PFLASH cannot be checked for “0” content by direct reads.

If the DFLASH is being operated in Complement Sensing Mode, this command verifies only that all read cells have sufficient high current that a program operation without prior erase is possible. After a certain operation history, a valid complement data entry may also appear as erased. Data qualifiers like headers or footers, which are usually used in EEPROM emulation, can be used to distinguish expected valid data from unknown data fields (see also [HF\\_ECCS.BLANKA](#) or [SF\\_ECCS.BLANKA](#)).

### Host Command Sequence Interpreter

The sequence error flag (DMU\_HF\_ERRSR.SQER) is set when the PA is not aligned to a page address.

The erase verify error flag (DMU\_HF\_ERRSR.EVER) is set to identify a found verification error.

See [Chapter 6.5.2.2.4](#) for Host Command Sequence Interpreter verify erase and analysis commands security.

### HSM Command Sequence Interpreter

The sequence error flag (DMU\_SF\_ERRSR.SQER) is set when the PA is not aligned to a page address.

The erase verify error flag (DMU\_SF\_ERRSR.EVER) is set to identify a found verification error.

If the DMU is in [Cranking Mode](#) or [Error Mode](#) then the command fails with an [Sequence Error \(SQER\)](#).

## Verify Erased WL

### Calling

- ST AA50<sub>H</sub>, WA
- ST AA58<sub>H</sub>, xx00<sub>H</sub>
- ST AAA8<sub>H</sub>, xx80<sub>H</sub>
- ST AAA8<sub>H</sub>, xx58<sub>H</sub>

### Function

This command verifies if one wordline addressed by “WA” is correctly erased, i.e. contain 0 data and ECC bits. Upon executing the command, the BUSY flag of the corresponding bank is set.

As the PFLASH has Safety ECC, the PFLASH cannot be checked for “0” content by direct reads.

If the DFLASH is being operated in Complement Sensing Mode, this command verifies only that all read cells have sufficient high current that a program operation without prior erase is possible. After a certain operation history, a valid complement data entry may also appear as erased. Data qualifiers like headers or footers, which are usually used in EEPROM emulation, can be used to distinguish expected valid data from unknown data fields (see also [HF\\_ECCS.BLANKA](#) or [SF\\_ECCS.BLANKA](#)).

### Host Command Sequence Interpreter

The sequence error flag (DMU\_HF\_ERRSR.SQER) is set when the WA is not aligned to a wordline address.

The erase verify error flag (DMU\_HF\_ERRSR.EVER) is set to identify a found verification error.

See [Chapter 6.5.2.2.4](#) for Host Command Sequence Interpreter verify erase and analysis commands security.

### HSM Command Sequence Interpreter

The sequence error flag (DMU\_SF\_ERRSR.SQER) is set when the WA is not aligned to a wordline address.

The erase verify error flag (DMU\_SF\_ERRSR.EVER) is set to identify a found verification error.

If the DMU is in [Cranking Mode](#) or [Error Mode](#) then the command fails with an [Sequence Error \(SQER\)](#).

### Verify Erased Logical Sector Range

#### Calling

- ST AA50<sub>H</sub>, SA
- ST AA58<sub>H</sub>, xxnn<sub>H</sub>
- ST AAA8<sub>H</sub>, xx80<sub>H</sub>
- ST AAA8<sub>H</sub>, xx5F<sub>H</sub>

#### Function

This command verifies if “nn” sectors starting at the sector addressed by “SA” are correctly erased, i.e. contain 0 data and ECC bits. Upon executing the command, the BUSY flag of the corresponding bank is set.

As the PFLASH has Safety ECC, the PFLASH cannot be checked for “0” content by direct reads.

If the DFLASH is being operated in Complement Sensing Mode, this command verifies only that all read cells have sufficient high current that a program operation without prior erase is possible. After a certain operation history, a valid complement data entry may also appear as erased. Data qualifiers like headers or footers, which are usually used in EEPROM emulation, can be used to distinguish expected valid data from unknown data fields (see also [HF\\_ECCS.BLANKA](#) or [SF\\_ECCS.BLANKA](#)).

### Host Command Sequence Interpreter

A sequence error (DMU\_HF\_ERRSR.SQER) is returned and the execution fails for the following conditions:

- The SA does not point to the base address of a correct sector or to an unavailable sector.
- The range of logical sectors is not contained in one physical sector.
- For DFLASH, if the range of logical sectors exceeds 256KByte addressable memory (256KByte physical memory in Single Ended mode, and 512KByte physical memory in Complement Sensing mode)
- For PFLASH, if the range of logical sectors exceeds 512KByte.
- For UCB, if the range of logical sectors exceeds 1.

The error flag (DMU\_HF\_ERRSR.EVER) is set to identify a found verification error.

See [Chapter 6.5.2.2.4](#) for Host Command Sequence Interpreter verify erase and analysis commands security.

### HSM Command Sequence Interpreter

A sequence error (DMU\_SF\_ERRSR.SQER) is returned and the execution fails for the following conditions:

- The SA does not point to the base address of a correct sector or to an unavailable sector.

The error flag (DMU\_SF\_ERRSR.EVER) is set to identify a found verification error.

If the DMU is in **Cranking Mode** or **Error Mode** then the command fails with an **Sequence Error (SQER)**.

### Erase Logical Sector Range

#### Calling

- ST AA50<sub>H</sub>, SA
- ST AA58<sub>H</sub>, xxnn<sub>H</sub>
- ST AAA8<sub>H</sub>, xx80<sub>H</sub>
- ST AAA8<sub>H</sub>, xx50<sub>H</sub>

#### Function

This command erases “nn” logical sectors starting at the sector addressed by “SA”. Upon executing the command, the BUSY flag of the corresponding bank is set.

A **Sequence Error (SQER)** is returned and the execution fails for the following conditions:

- The range of logical sectors is not contained in one physical sector.
- For PFLASH, if the range of logical sectors exceeds 512KByte.
- For DFLASH, if the range of logical sectors exceeds 256KByte addressable memory (256KByte physical memory in Single Ended mode, and 512KByte physical memory in Complement Sensing mode)
- For UCB, if the range of logical sectors exceeds 1.
- If SA does not align to the start address of a logical sector.
- If SA aligns to an unavailable sector.
- If “nn” logical sectors is set to 0.
- If the DMU is in **Cranking Mode** or **Error Mode**.

A **Protection Error (PROER)** is returned and the execution fails for the following conditions:

- If SA or any of the following nn-1 logical sectors has an active write protection.
- The Flash module has an active global read protection set.

The error flag EVER is set to indicate an error during the erase process.

The flag PVER is set when the erase counter programming incurs an error.

**Note:** *The duration of an erase command can be much shorter than documented ( $t_{ERD}$ ,  $t_{MERD}$ ,  $t_{ERP}$ ,  $t_{MERP}$ ) when the range is already erased or partly erased.*

### Resume NVM Operation

### **Calling**

- ST AA50<sub>H</sub>, PA/SA
- ST AA58<sub>H</sub>, xxnn<sub>H</sub>
- ST AAA8<sub>H</sub>, xx70<sub>H</sub>
- ST AAA8<sub>H</sub>, xxCC<sub>H</sub>

### **Function**

A suspended command may be resumed.

### **Disable Protection**

### **Calling**

- ST 553C<sub>H</sub>, UC
- ST.W 553C<sub>H</sub>, PW0
- ST.W 553C<sub>H</sub>, PW1
- ST.W 553C<sub>H</sub>, PW2
- ST.W 553C<sub>H</sub>, PW3
- ST.W 553C<sub>H</sub>, PW4
- ST.W 553C<sub>H</sub>, PW5
- ST.W 553C<sub>H</sub>, PW6
- ST.W 553C<sub>H</sub>, PW7

### **Function**

The password protection of the selected UCB (if this UCB offers this feature) is temporarily disabled by setting DMU\_HF\_PROTECT.PRODISx (with “x” indicating the UCB) or DMU\_HF\_PROTECT.SRT when all the passwords PW0–PW7 match their configured values in the corresponding UCB.

The command fails by setting PROER when any of the supplied PWs does not match. In this case until the next application reset all further calls of **Disable Protection** fail with PROER independent of the supplied password.

### **Host Command Sequence Interpreter**

The protection handling has to use the Host command interface.

### **HSM Command Sequence Interpreter**

This command is rejected by the HSM command sequence interpreter with DMU\_SF\_ERRSR.SQER.

### **Resume Protection**

### **Calling**

- ST 5554<sub>H</sub>, xxF5<sub>H</sub>

## Function

This command clears all DMU\_HF\_PROTECT.PRODISx and DMU\_HF\_PROTECT.SRT effectively enabling again the Flash protection as it was configured.

### Host Command Sequence Interpreter

The protection handling has to use the Host command interface.

### HSM Command Sequence Interpreter

This command is rejected by the HSM command sequence interpreter with DMU\_SF\_ERRSR.SQER.

## Clear Status

### Calling

- ST 5554<sub>H</sub>, xxFA<sub>H</sub>

### Function

Clear operation and error flags.

Clearing of error flags will not solve the underlying problem that caused the error flag.

### Host Command Sequence Interpreter

The following flags are cleared:

- Operation flags: DMU\_HF\_OPERATION.PROG, DMU\_HF\_OPERATION.ERASE.
- Error flags: DMU\_HF\_ERRSR.SQER, DMU\_HF\_ERRSR.PROER, DMU\_HF\_ERRSR.PVER and DMU\_HF\_ERRSR.EVER.

### HSM Command Sequence Interpreter

The following flags are cleared:

- Operation flags: DMU\_SF\_OPERATION.PROG, DMU\_SF\_OPERATION.ERASE.
- Error flags: DMU\_SF\_ERRSR.SQER, DMU\_SF\_ERRSR.PVER and DMU\_SF\_ERRSR.EVER.

## 6.5.2.2.4 Protection for Verify Command Sequences

For the Host Command Sequence Interpreter, the command sequence function may be limited by configuring DMU\_SP\_PROCONHSMCFG.BLKFLAN = 1<sub>B</sub>:

If DMU\_SP\_PROCONHSMCFG.BLKFLAN = 0<sub>B</sub> then function is enabled for every address.

If DMU\_SP\_PROCONHSMCFG.BLKFLAN = 1<sub>B</sub> then function is blocked as follows:

- PFLASH PF0 logical sector analysis is blocked if DMU\_SP\_PROCONHSMCX0.HSMxX = 1<sub>B</sub> or DMU\_SP\_PROCONHSMCX1.HSMxX = 1<sub>B</sub> for the corresponding PFLASH HSM code logical sector "s".
- DFLASH1 analysis is blocked if DMU\_SP\_PROCONHSMCFG.HSMDX = 1<sub>B</sub>.
- UCB\_HSMCFG analysis is blocked if DMU\_HF\_CONFIRM0.PROINHSMCFG = **CONFIRMED** or **ERRORED**.
- UCB\_RETEST analysis is blocked if DMU\_HF\_CONFIRM0.PROINSRT = **CONFIRMED** or **ERRORED**.
- UCB\_PFLASH\_ORIG and UCB\_PFLASH\_COPY analysis is blocked if one of the following conditions is true:
  - The UCB\_PFLASH confirmation state is **CONFIRMED** and **Disable Protection** has not been activated.
  - The UCB\_PFLASH confirmation state is **ERRORED**.

- UCB\_DFLASH\_ORIG and UCB\_DFLASH\_COPY analysis is blocked if one of the following conditions is true:
  - The UCB\_DFLASH confirmation state is **CONFIRMED** and **Disable Protection** has not been activated.
  - The UCB\_DFLASH confirmation state is **ERRORED**.
- UCB\_DBG\_ORIG and UCB\_DBG\_COPY is blocked if one of the following conditions is true:
  - The UCB\_DBG confirmation state is **CONFIRMED** and **Disable Protection** has not been activated.
  - The UCB\_DBG confirmation state is **ERRORED**.
- UCB\_ECPPIO\_ORIG and UCB\_ECPPIO\_COPY analysis is blocked if one of the following conditions is true:
  - The UCB\_ECPPIO confirmation state is **CONFIRMED** and **Disable Protection** has not been activated.
  - The UCB\_ECPPIO confirmation state is **ERRORED**.
- UCB\_BMHDX\_ORIG and UCB\_BMHDX\_COPY analysis is blocked if one of the following conditions is true:
  - The UCB\_BMHDX confirmation state is **CONFIRMED** and **Disable Protection** has not been activated.
  - The UCB\_BMHDX confirmation state is **ERRORED**.

If this function is blocked the command fails by setting DMU\_HF\_ERRSR.PROER

### 6.5.2.2.5 DMU Commands

The DMU implements the following commands:

- **Reset to Read**
- **Clear Status**
- **Disable Protection**
- **Resume Protection**

### 6.5.2.2.6 Suspend and Resume Operations

The following operations may be suspended:

- **Write Page, Write Page Once, Write Burst, Write Burst Once.**
  - For **Write Page** and **Write Page Once**, the write operation may be suspended between the end of the programming and the start of the verification. Therefore the programming process itself is not suspended
  - For **Write Burst** and **Write Burst Once**, the burst operation in PFLASH may be suspended during the programming process. In DFLASH it is only suspended after end of the programming and before the start of verification.
- **Erase Logical Sector Range.**
- **Verify Erased Page, Verify Erased WL, Verify Erased Logical Sector Range.**
- **Replace Logical Sector**

There can be at most one operation in the suspended state in the **Host Command Interface** and one in the **HSM Command Interface**.

Erroneous suspend and resume requests are detected and prevented by the DMU.

The target range of a suspend program or erase operation is in an undefined state. Reading this range delivers unpredictable results.

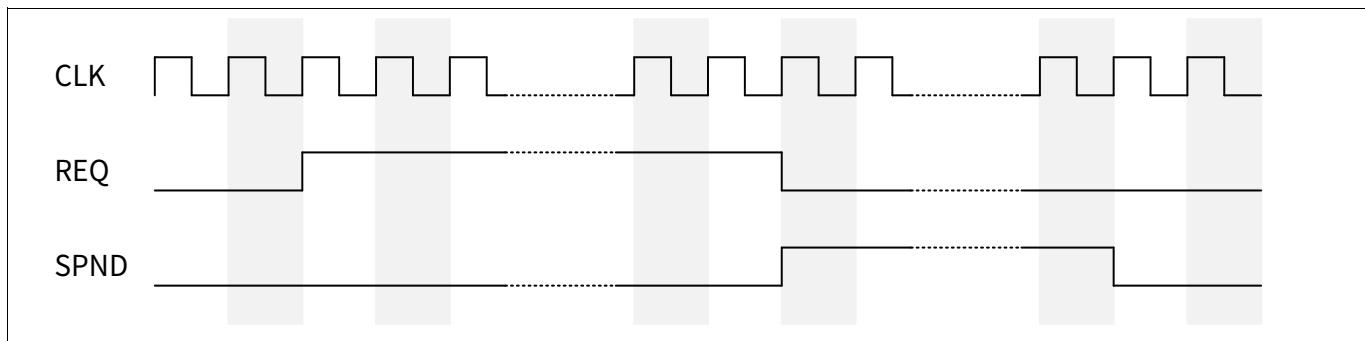
#### Host Command Interface

The Host Command Interface may suspend one flash operation (at [Figure 62](#)).

#### Suspend

Software may suspend a flash operation by writing  $1_B$  to DMU\_HF\_SUSPEND.REQ.

- The **Host Command Interface** checks if there is a flash operation requested or ongoing
  - If not then the suspend request is ignored and DMU\_HF\_SUSPEND.REQ is cleared.
- The **Host Command Interface** checks if the ongoing flash operation is suspendable.
  - If not then the DMU waits for the flash operation to complete and then clears DMU\_HF\_SUSPEND.REQ
  - If suspendable, the **Host Command Interface** checks if there is already a suspended operation. If true then a suspend error is reported (DMU\_HF\_SUSPEND.ERR = 1<sub>B</sub>) and the request is cleared (DMU\_HF\_SUSPEND.REQ = 0<sub>B</sub>).
- Else the **Host Command Interface** suspend request is serviced by the DMU.
- The DMU waits for the flash operation to reach an interruptible state and then clears the busy flag. The **Host Command Interface** reports the following (at **Figure 61**):
  - Operation suspended: DMU\_HF\_SUSPEND.SPND is set and DMU\_HF\_SUSPEND.REQ is cleared. The DMU stores which process was suspended and its arguments.
  - Operation finished: DMU\_HF\_SUSPEND.REQ is cleared.



**Figure 61 Operation Suspended and Operation Finished**

## Resume

A suspended operation may be resumed with the command sequence **Resume NVM Operation**.

- The DMU checks if there is a suspended operation.
  - If not then a **Sequence Error (SQER)** is reported.
- The arguments “PA” or “SA” and “nn” of the **Resume NVM Operation** must be identical to the argument of the suspended command. Using a different argument is detected by the DMU and lets the resume fail with a **Sequence Error (SQER)** and DMU\_HF\_SUSPEND.SPND remains set.
- Else for a valid **Resume NVM Operation** the DMU sets corresponding bank busy flag, clears the suspend flag (DMU\_HF\_SUSPEND.SPND = 0<sub>B</sub>) and sets appropriate status flag DMU\_HF\_OPERATION.{PROG/ERASE}.

**Attention:** Please ensure that between the start or resume of a Flash operation and the suspend request normally at least ~3ms execution time can pass. This is especially important for Erase and Erase Verify Operations. Shorter execution durations are possible but the erase/verify process will not advance at all. To avoid a high number of repetitions and very long total execution timings it is recommended to wait at least in average ~10ms. Timings mentioned above are valid for f(FSI) = 100 MHz and need to be scaled up accordingly when using a lower FSI frequency.

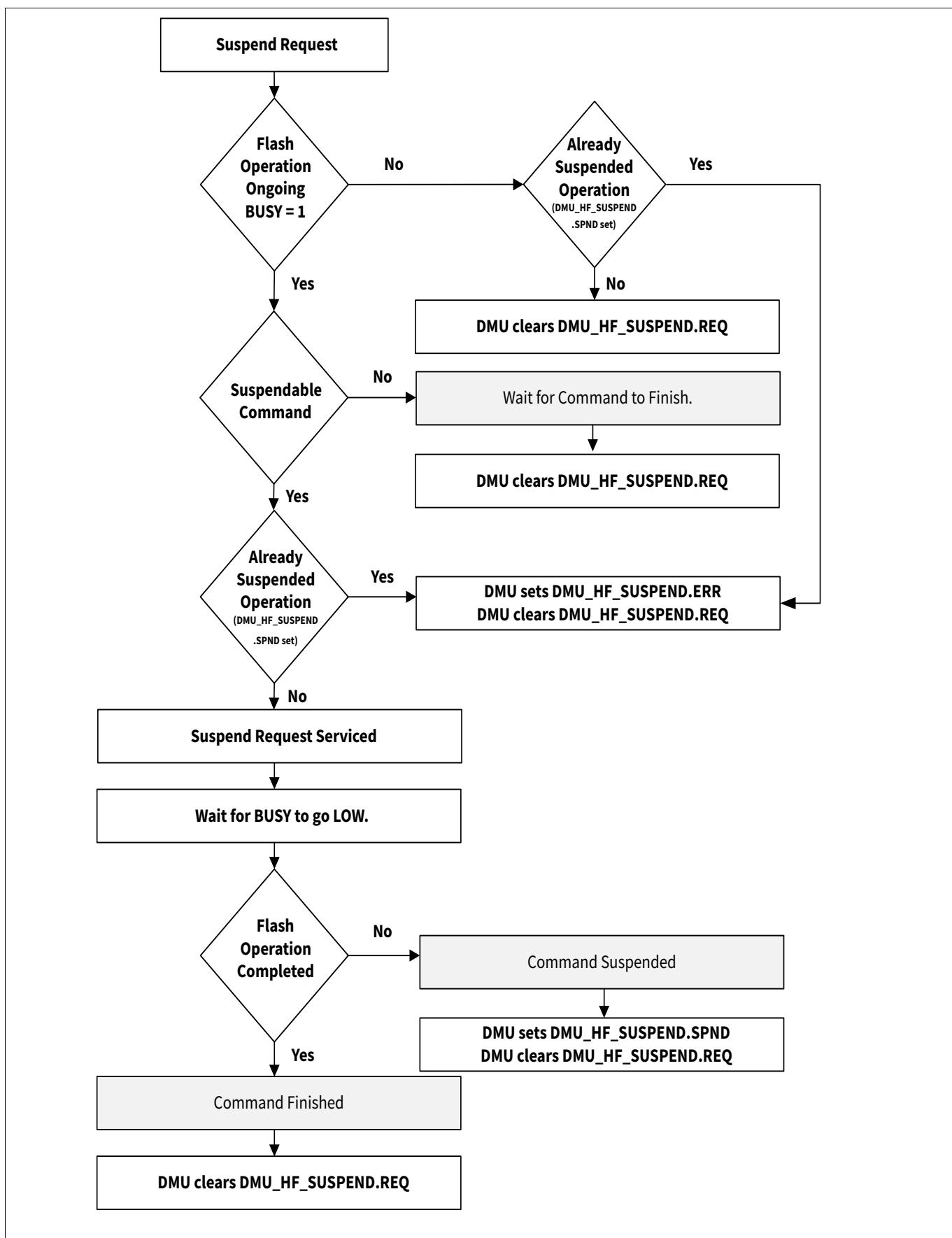


Figure 62 Suspend

## HSM Command Interface

The HSM Command Interface suspend and resume operation is like that of the [Host Command Interface](#).

The DMU\_SF\_SUSPEND register is used to control and monitor suspend operations.

## Suspended States

### Suspended Programming State

In the suspended programming state:

- Reading Flash is allowed.
- New programming commands are rejected with SQER:
  - [Enter Page Mode](#).
  - [Write Page](#), [Write Page Once](#), [Write Burst](#), [Write Burst Once](#).
  - [Replace Logical Sector](#)
- New erase and verify commands are rejected with SQER:
  - [Erase Logical Sector Range](#).
  - [Verify Erased Page](#), [Verify Erased WL](#), [Verify Erased Logical Sector Range](#).

### Suspended Erase State

In the suspended erase state:

- Reading Flash is allowed.
- New programming commands may be performed on any bank:
  - [Enter Page Mode](#).
  - [Write Page](#), [Write Page Once](#), [Write Burst](#), [Write Burst Once](#).
  - [Replace Logical Sector](#)
- However programming in the target range of the suspended erase fails with SQER.
- New erase and verify commands are rejected with SQER:
  - [Erase Logical Sector Range](#).
  - [Verify Erased Page](#), [Verify Erased WL](#), [Verify Erased Logical Sector Range](#).

### Suspended Verify State

In the suspended [Verify Erased Page](#), [Verify Erased WL](#) or [Verify Erased Logical Sector Range](#) state:

- Reading Flash is allowed.
- New programming commands may be performed on any bank:
  - [Enter Page Mode](#).
  - [Write Page](#), [Write Page Once](#), [Write Burst](#), [Write Burst Once](#).
  - [Replace Logical Sector](#)
- New erase and verify commands are rejected with SQER:
  - [Erase Logical Sector Range](#).
  - [Verify Erased Page](#), [Verify Erased WL](#), [Verify Erased Logical Sector Range](#).

## Suspend-Resume application note for Host Command Interface

The following notes provide guidance in using the suspend and resume feature for Host Command Interface

In the case of a request for suspending an ongoing NVM operation:

- Please ensure that there is a delay between start or resume of an erase process and the suspend request. For further details on the delay, please refer to the ‘Attention’ section of [Chapter](#).
- Check if the corresponding BUSY flag has already been cleared. If yes, no suspend is necessary.
- Check DMU\_HF\_SUSPEND.SPND bit. If this is ‘1’, then there is already a suspended operation and the ongoing operation cannot be suspended.
- Request suspend with the control flag DMU\_HF\_SUSPEND.REQ.
- Wait until the corresponding BUSY flag clears.
- After that, check the DMU\_HF\_SUSPEND.SPND bit. If this is ‘1’, then the operation was suspended and needs to be resumed later. If this is ‘0’, the operation has already finished, therefore, no resume is necessary.
- Now, new Flash operations are allowed with the restrictions documented in [Chapter](#).

The resume of the NVM operation is done in these steps:

- Check if DMU\_HF\_SUSPEND.SPND is set. If this is ‘1’, there is an operation suspended that can be resumed.
- Resume the operation with the command sequence “Resume NVM Operation”.
- Wait until the DMU\_HF\_SUSPEND is ‘0’.
- After that, wait for the end of operation signalled by the corresponding BUSY flag going to ‘0’.

The same guidance is applicable for HSM Command Interface using the DMU\_SF\_SUSPEND register.

### 6.5.2.2.7 Programming Voltage Selection

If a device supports an external 5V supply via  $V_{EXT}$  then DMU\_HF\_PCONTROL.PR5V may be used to select an external supply as the source of the programming voltage:

- DMU\_HF\_PCONTROL.PR5V set to “P5V”
  - The programming voltage is sourced directly from the external supply  $V_{EXT}$  supplied with a nominal 5V.
  - See data sheet for supply voltage tolerances.
- DMU\_HF\_PCONTROL.PR5V set to “P3V”
  - The programming voltage is internally generated from  $V_{DDP3}$  supplied with a nominal 3.3 V.
  - The allowed range of  $V_{EXT}$  is much larger in this configuration.

For the PFLASH the P5V mode offers significantly shorter programming times (see data sheet).

### 6.5.2.2.8 Performing Flash Operations

This section offers advice for using command sequences.

#### General Advice

- Please remember to disable the Safety ENDINIT protection before executing program, erase and user commands for the PFLASH.
- Code that performs PFLASH programming or erasing should not be executed from the same PFLASH.

- Command cycles shall address the non-cached address range of the Flash (otherwise the data may stay in the cache or could be received by the DMU in an incorrect order).
- The prefetch buffers and data line buffer used for PFLASH reads are automatically invalidated after changing PFLASH content by erasing or programming. The prefetch buffers can additionally be invalidated by writing **HF\_PCONTROL.DEMAND** to '11', disabling prefetch. All buffers are invalidated by a reset.
- The NVM Subsystem works with the SRI, FSI and FSI2 clocks. When changing the divider values of these clocks in SCU\_CCUCON0 the following rules apply:
  - The only allowed DMU/Flash “operation” when switching is reading from Flash memory. Any other operation on the Flash is forbidden.
  - It must be ensured that before, during and after the divider change the configured number of Flash wait-cycles is sufficient for the selected clock frequency. Remember that the wait-cycles are counted with fFSI2 for PFlash and fFSI for DFlash.
  - fFSI2 must not be programmed slower than fSRI.
  - After System Resets and Power-On Resets the wait cycle values are configured to a value only sufficient for the clock frequencies used during startup (i.e. 100 MHz). So basically always changes to the clock configuration must be preceded by changes to the wait cycles.

### Flushing the CPU buffer when using NVM commands

The NVM **Load Page** command requires that all write data transfers for one page must have the same width (either all 32-bit or all 64-bit). If 32-bit **Load Page** commands are used there would be pairs of two writes as follows:

```
ST 0x55F0
ST 0x55F4
```

```
ST 0x55F0
ST 0x55F4
```

```
ST 0x55F0
ST 0x55F4
```

If the CPU Store function is active, some of these 32-bit writes might be merged into a 64-bit write. In order to prevent this and have all **Load Page** commands in the same order it is necessary to place a DSYNC instruction after every 32-bit write (or a register read, for example to **HF\_ID**, which would have the same effect). This ensures that all data is written to the memory prior to the execution of the next ST write instruction. Thus the above program sequence would become (with an additional DSYNC at the start of the sequence to ensure there is no residual operation lying around in the store buffer):

```
DSYNC
ST 0x55F0
DSYNC
ST 0x55F4
DSYNC
```

```
DSYNC
ST 0x55F0
DSYNC
ST 0x55F4
DSYNC
```

```
DSYNC
```

```
ST 0x55F0
DSYNC
ST 0x55F4
DSYNC
```

It is recommended to insert a DSYNC (or register read) after each program command sequence to ensure the correct sequencing of commands to the DMU.

### Sequence for Programming

The following sequence is the most defensive one for programming a page. It is however acceptable to skip some checks when the programmed data is later verified:

- “Clear Status” to clear flags.
- “Enter Page Mode”.
- DSYNC.
- Wait until **HF\_STATUS.PFPAGE** = ‘1’ or **HF\_STATUS.DFPAGE** = ‘1’ depending upon the Flash target, or fail if **HF\_ERRSR.SQER** = ‘1’ or **HF\_ERRSR.PROER** = ‘1’.
- Repeat “Load Page” until the page is filled.
- “Write Page”.
- DSYNC.
- Wait until **HF\_OPERATION.PROG** = ‘1’ or fail if (**HF\_ERRSR.SQER** = ‘1’ or **HF\_ERRSR.PROER** = ‘1’).
- Wait for 2\*1/fFSI ns (DFlash) or 3\*1/fFSI + 8\*1/fSRI ns (PFlash).
- Wait until **HF\_STATUS.xBUSY** = ‘0’ or enable the interrupt.
  - While **HF\_STATUS.xBUSY** is ‘1’ the flags **HF\_ERRSR.OPER** can be checked for ‘1’ as abort criterion to protect against hardware failures causing BUSY to stay ‘1’.
- Check for **HF\_ERRSR.PVER** flag.
- Fail if **HF\_ERRSR.OPER** = ‘1’.
- Recommended: check programmed content, evaluate **HF\_ECCS** for DFlash accesses, and possibly count correctable errors.
- Clear error flags either with “Clear Status” or by directly writing to **HF\_CLRE**.

### Sequence for Erasing

The following sequence is the most defensive one for erasing a range of sectors. It is however acceptable to skip some checks when the programmed data is verified after programming (or the flag **HF\_ERRSR.PVER** is checked as described above):

- “Clear Status” to clear flags
- “Erase Logical Sector Range”.
- DSYNC.
- Wait until **HF\_OPERATION.ERASE** = ‘1’ or fail if (**HF\_ERRSR.SQER** = ‘1’ or **HF\_ERRSR.PROER** = ‘1’).
- Wait for 2\*1/fFSI ns (DFlash) or 3\*1/fFSI + 8\*1/fSRI ns (PFlash).
- Wait until **HF\_STATUS.xBUSY** = ‘0’ or enable the interrupt.
  - While **HF\_STATUS.xBUSY** is ‘1’ the flags **HF\_ERRSR.OPER** can be checked for ‘1’ as abort criterion to protect against hardware failures causing BUSY to stay ‘1’.
- Check for **HF\_ERRSR.PVER** and **HF\_ERRSR.EVER** flags.
- Fail if **HF\_ERRSR.OPER** = ‘1’.

- Clear error flags either with “Clear Status” or by directly writing to **HF\_CLRE**.

An analog sequence can be used for “Verify Erased Logical Sector Range”.

### Resets during Flash Operation

A reset or power failure during an ongoing Flash operation (i.e. program or erase) must be considered as a violation of stable operating conditions. However, the Flash was designed to prevent damage to non-addressed Flash ranges when the reset is applied as defined in the data sheet. The addressed Flash range is left in an undefined state.

### General advice

When an erase operation is aborted the previously programmed bits ('1') in the addressed Flash range can be in any state between '0' and '1'. When reading this range all-0 can be returned, the old data, or something in between. The result can be unstable. Due to the ECC correction there may even appear '1' bits at positions which contained '0' bits before erase start.

When a page programming operation is aborted the page can still appear as erased (but contain slightly programmed bits), it can appear as being correctly programmed (but the data has a lowered retention) or the page contains garbage data. It is also possible that the read data is unstable so that depending on the operating conditions different data is read.

For the detection of an aborted Flash process the flags **HF\_OPERATION.PROG**, **HF\_OPERATION.ERASE** and **HF\_OPERATION.USER** could be used as an indicator, but only when the reset was an application reset. When Flash processes are aborted by power-on resets, this is not indicated by any flags. It is not possible to detect an aborted operation simply by reading the Flash range (please note that the ECC is not a reliable means to detect an aborted Flash operation). Even the margin reads don't offer a reliable indication. When erasing or programming the PFlash usually an external instance can notice the reset and restart the operation by erasing the Flash range and programming it again.

### Advice for EEPROM Emulation

However for the case of EEPROM emulation in the DFlash this external instance is not existing. A common solution is detecting an abort by performing two operations in sequence and determine after reset from the correctness of the second the completeness of the first operation, for example, after erasing a DFlash sector a page is programmed. After reset the existence of correct data in this page proves that the erase process was performed completely. The detection of aborted programming processes can be handled similarly. After programming a block of data an additional page is programmed as marker. When after reset the block of data is readable and the marker is existent it is ensured that the block of data was programmed without interruption. In very specific cases it is allowed to repair data left from an aborted programming operation: if the algorithm can detect that an abort occurred and the algorithm knows which data must be present in the page it is possible to simply redo the programming by programming the same data again. In the AURIX2G family the probability of aborting a programming process can be minimized by the following means:

- In case of a reset with stable power supply (“warm resets”) the Flash gets automatically a request to enter shutdown state before the reset is applied. Due to their short duration, single “Write Page” operations are finished correctly by this process. “Write Burst” operations however are interrupted after the current page programming has finished.
- The voltage monitoring can be used to get an under voltage warning early enough that an ongoing programming process can be finished and no new one is started.
- By selecting an internally generated programming voltage, the needed voltage at VEXT is reduced giving more headroom for an early warning by the voltage monitors.

### 6.5.2.3 Traps

Generally the DMU peripherals report fatal errors by issuing a bus error which is translated by the CPU into a trap.

The conditions for reporting a bus error are:

- Uncorrectable ECC error.
- Write access to read-only register.
- Write access to Boot ROM (BROM).
- Write access to an access controlled register or Flash address range by a master without allowance by the register access protection.
- Not allowed write access to protected register (e.g. SV, Endinit or Safety Endinit).
- Not allowed Flash read access with active read protection.
- Read and write access to the FSI when the DMU is in sleep mode.
- Read access to not available Flash memory.
- Write access to not available Flash memory.
- Read-modify-write access to the Flash memory.
- Read access to a busy PFLASH bank (if not disabled by with CPUx\_FLASHCON1.STALL).
- Read access to a busy DFLASH bank (DFLASH0/1 EEPROM, CFS or UCB)
- Block Transfer to a DFLASH bank.
- Write access to the DFLASH0 address range when the Host Command Interface is busy with a command.
- Write access to the DFLASH1 address range when the HSM Command Interface is busy with a command.
- Read or write access to unoccupied register address.

### 6.5.2.4 Interrupts

#### 6.5.2.4.1 Host Command Interface

The following events can trigger an interrupt service request to the Interrupt Router (IR):

- End of BUSY: if DMU\_HF\_EER.EOBM = 1<sub>B</sub> and one of the DMU\_HF\_STATUS flags D0BUSY, D1BUSY or PFLASH flags transitions from ‘1’ to ‘0’ then an interrupt service request is triggered (e.g. wake-up, erase sequences or program sequences).
- **Operation Error (OPER):** if DMU\_HF\_EER.OPERM = 1<sub>B</sub> and DMU\_HF\_ERRSR.OPER flag is set.
- **Protection Error (PROER):** if DMU\_HF\_EER.PROERM = 1<sub>B</sub> and DMU\_HF\_ERRSR.PROER flag is set.
- **Sequence Error (SQER):** if DMU\_HF\_EER.SQERM = 1<sub>B</sub> and DMU\_HF\_ERRSR.SQER flag is set.
- **Erase Verify Error (EVER):** if DMU\_HF\_EER.EVERM = 1<sub>B</sub> and DMU\_HF\_ERRSR.EVER flag is set.
- **Program Verify Error (PVER):** if DMU\_HF\_EER.PVERM = 1<sub>B</sub> and DMU\_HF\_ERRSR.PVER flag is set.

The event that triggered the interrupt can be determined from the DMU\_HF\_STATUS and DMU\_HF\_ERRSR registers.

An interrupt event must be triggered when the event appears again and the corresponding status flag is still set. End of BUSY interrupts are only generated after completion of startup.

#### 6.5.2.4.2 HSM Command Interface

The following events can trigger an application interrupt to the HSM module:

- End of BUSY: if DMU\_SF\_EER.EOBM = 1<sub>B</sub> and HSM flag D1BUSY transitions from ‘1’ to ‘0’ then an interrupt service request is triggered (e.g. wake-up, erase sequences or program sequences).
- **Operation Error (OPER)**: if DMU\_SF\_EER.OPERM = 1<sub>B</sub> and DMU\_SF\_ERRSR.OPER flag is set.
- **Sequence Error (SQER)**: if DMU\_SF\_EER.SQERM = 1<sub>B</sub> and DMU\_SF\_ERRSR.SQER flag is set.
- **Erase Verify Error (EVER)**: if DMU\_SF\_EER.EVERM = 1<sub>B</sub> and DMU\_SF\_ERRSR.EVER flag is set.
- **Program Verify Error (PVER)**: if DMU\_SF\_EER.PVERM = 1<sub>B</sub> and DMU\_SF\_ERRSR.PVER flag is set.

The event that triggered the interrupt can be determined from the DMU\_SF\_STATUS and DMU\_SF\_ERRSR registers.

An end of BUSY interrupt is only generated after completion of startup.

### 6.5.2.5 Error Handling

Customer software should handle errors during startup and operation as follows:

#### 6.5.2.5.1 Handling Errors During Startup

The status flags are not only used to inform about the success of Flash command sequences but they are also used to inform (1) the startup software and (2) the user software about special situations incurred during startup. In order to react on this information these flags must be evaluated after reset and before performing any flag clearing sequences such as **Reset to Read** or **Clear Status**.

The following two levels of situations are separated:

- Fatal level: the user software is not started. A WatchDog Timer (WDT) reset is performed.
- Warning level: the user software is started but a warning is issued.

##### Fatal Level (WDT Reset)

These error conditions are evaluated by the startup software which decides that the Flash is not operable and thus waits for a WDT reset. The application sees only a longer startup time followed by a WDT reset.

The reason for a failed Flash startup can be a hardware error or damaged configuration data.

##### Warning Level

These conditions inform the user software about an internally corrected or past error condition.

Leftover OPER: FSI\_ERR.OPERERR

Status bits set: DMU\_HF\_ERRSR.OPER

The OPER flag is only cleared by a system reset. After any other reset a OPER flag is still set when the user software is started.

#### 6.5.2.5.2 Handling Errors During Operation

During operation (i.e. after issuing command sequences) error conditions are handled as follows:

##### Sequence Error (SQER)

Fault conditions:

- Improper command cycle address or data, i.e. incorrect command sequence.
- New **Enter Page Mode** in **Page Mode**.
- **Load Page** and not in **Page Mode**.
- **Load Page** with mixed 32/64-bit transfers.

- **Load Page** with invalid operation code - operation code must be SDTD or SDTW.
- For a 32-bit transfer the first **Load Page** addresses the upper 32-bit word.
- Write commands with incorrect buffer sizes.
- Write commands not in **Page Mode**.
- All commands to unavailable Flash range (e.g. Flash range does not physically exist).
- Command sequence with address not aligned to a legal start address (e.g. page, UCB or sector).
- Command sequence not pointing to a PFLASH, DFLASH0\_EEPROM, DFLASH0\_UCB or DFLASH1\_EEPROM address.
- **Write Page Once** targeting DFLASH.
- **Write Burst Once** targeting DFLASH.
- **Erase Logical Sector Range** or **Verify Erased Logical Sector Range** with range leaving a physical sector.
- **Resume NVM Operation** with arguments not matching the suspended command.
- **Resume NVM Operation** when there is no suspended operation.
- Any programming or erase command when there is a suspended programming command.
- Any erase command when there is a suspended erase command, including **Verify Erased WL** and **Verify Erased Logical Sector Range**.
- Programming to the target range of a suspended erase command.
- Unsupported command sequence for **Error Mode**.
- Unsupported command sequence for **Cranking Mode**.
- Unsupported command sequence for the HSM command sequence interpreter.
- Illegal value of SCU\_SWAPCTRL bits when DMU\_HP\_PROCONTP.SWAPEN is “Enabled” and a write or erase command is requested.

New state:

The command interface enters the idle state with following exceptions:

- **Enter Page Mode** in **Page Mode** re-enters **Page Mode**.
- **Write Page**, **Write Page Once**, **Write Burst** or **Write Burst Once** with buffer underflow is executed.
- After **Load Page** causing a buffer overflow the **Page Mode** is not left, a following **Write Page**, **Write Page Once**, **Write Burst** or **Write Burst Once** is executed.

Proposed handling by software:

Usually this bit is only set due to a bug in the software. Therefore in development code the responsible error tracer should be notified. In production code this error will not occur. It is however possible to clear this flag with **Clear Status** or **Reset to Read** and simply issue the corrected command sequence again.

## Operation Error (OPER)

Fault conditions:

The FSI may report an OPER at any time. Possible causes include:

- Double-bit ECC error detected while executing microcode out of FSI SRAM.
- Transient event due to alpha-particles or illegal operating conditions.
- Permanent error due to a hardware defect

New state:

The Flash operation is aborted and **Error Mode** is entered.

Proposed handling by software:

The last operation can be determined from the PROG and ERASE flags. The PROG or ERASE flag should be cleared with **Clear Status**. In case of an erase operation the affected physical sector must be assumed to be in an invalid state, in case of a program operation only the affected page. Other physical sectors can still be read. New program or erase commands must not be issued before the next reset.

A system reset must be applied to perform a new Flash startup with initialization of the FSI SRAM and clear the OPER flag. The application must determine from the context which operation failed and react accordingly. Mostly erasing the addressed sector and re-programming its data is most appropriate. If a **Write Page**, **Write Page Once**, **Write Burst** or **Write Burst Once** command was affected and the sector can not be erased (e.g. in Flash EEPROM emulation) the wordline could be invalidated if needed by marking it with all-one data and the data could be programmed to another empty wordline.

Only in case of a defective FSI SRAM the next program or erase operation will incur again this error.

**Note:** *Although an OPER indicates a failed operation it is possible to ignore it and rely on a data verification step to determine if the Flash memory has correct data. Before re-programming the Flash the flow must ensure that a new reset is applied.*

### Protection Error (PROER)

Fault conditions:

- Password failure.
- Erase/Write to protected sector.
- Replace Logical Sector command to a protected sector.
- **Erase Logical Sector Range** of UCB with active protection.
- Write commands to UCB with active protection.
- Verify commands to blocked addresses:
  - **Verify Erased Page**, **Verify Erased WL**, **Verify Erased Logical Sector Range**.
- Program, erase and replace logical sector commands targeting PFLASH with active Safety ENDINIT protection.
- During rampup if for single and dual UCB both the ORIG and COPY confirmation codes are **ERRORED**.

New state:

The command interface enters the idle state. The protection violating command is not executed.

Proposed handling by software:

Usually this bit is only set during runtime due to a bug in the software. In case of a password failure or UCB error a reset must be performed in the other cases the flag can be cleared with **Clear Status**. After that the corrected sequence can be executed.

### Erase Verify Error (EVER)

Fault conditions:

This flag is set by the erase commands when they don't achieve an optimum result. This is also set if a verification error is found by the verify erase commands or if there is no free entry available when performing **Replace Logical Sector**.

New state:

No state change. Just the bit is set.

Proposed handling by software:

This bit should be cleared with **Clear Status**.

The operating conditions should be checked. The following advice assumes correct operating conditions and a correctly configured device.

In the PFLASH an EVER indicates a fail in a logical sector. The command **Replace Logical Sector** can be used to replace that logical sector with a free redundant logical sector.

In the DFLASH an EVER indicates a potential fail at a wordline. It is recommended to repeat the erase once, however, if the EVER still persists, the robust EEPROM emulation has to “jump” over this wordline and program its data to the next wordline.

**Note:** *Even when this flag is ignored it is recommended to clear it at the end of the command, including suspend or resume of the command. Otherwise all following operations – including “sleep” – could trigger an interrupt even when they are successful.*

### **Program Verify Error (PVER)**

Fault conditions:

This flag is set by the program commands when they don't achieve an optimum result. This is also set during the execution of an erase command when the erase counter programming encounters an error or if there is a failure during the programming of the redundancy information while performing **Replace Logical Sector**.

New state:

No state change. Just the bit is set.

Proposed handling by software:

This bit should be cleared with **Clear Status**.

The operating conditions should be checked. The following advice assumes correct operating conditions and a correctly configured device.

In the PFLASH a PVER could indicate a fail in a logical sector. The command **Replace Logical Sector** can also be used to replace a failing logical sector with a free redundant logical sector.

In the DFLASH a PVER is a signal for the robust EEPROM emulation that programming on the current wordline failed. It is recommended to repeat the programming once, however, if the PVER still persists, the algorithm has to “jump” over this wordline and program its data to the next wordline.

**Note:** *Even when this flag is ignored it is recommended to clear it at the end of the command, including suspend or resume of the command. Otherwise all following operations – including “sleep” – could trigger an interrupt even when they are successful.*

### **Original Error (ORIER)**

Fault conditions:

This flag is set when a UCB ORIG confirmation code is **ERRORED** in a UCB containing both ORIG and COPY confirmation codes.

Proposed handling:

This bit should be cleared with application reset.

### **6.5.2.6 DMU Modes**

The DMU functions in one of the following modes (**Figure 63**)

### 6.5.2.6.1 Operation Mode

The PFLASH and DFLASH modules are powered up. The NVM may be read and the DMU must interpret command sequences. If a flash operation is performed then the response to a read access to a busy PFLASH bank is determined by the stall function:

- Error response ( $\text{STALL} = 0_B$ ): bus error.
- Stall response ( $\text{STALL} = 1_B$ ): ready is suppressed until BUSY clears.

Read access to any section of a busy DFLASH bank (either EEPROM, UCB or CFS) always generates a bus error.

Control of the power sub-modes is:

- **Demand Mode:** flash prefetch buffer accesses may be disabled by software.
- **Dynamic Idle Mode:** may be configured by software.

The following programming program sub-mode is supported:

- **Page Mode:** flash modules enabled to received data for programming.

The PFLASH read cycles are calculated from DMU\_HF\_PWAIT.RFLASH and DMU\_HF\_PWAIT.RECC

### 6.5.2.6.2 Error Mode

If the FSI has reported an **Operation Error (OPER)** in **Operation Mode** or **Cranking Mode** then the DMU is also in **Error Mode**. An on-going flash operation is aborted and the BUSY flag is cleared. Flash reads are not impaired.

Only the **DMU Commands** and FSI ROM Commands (see FSI chapter for details) are supported. The remaining commands fail with **Sequence Error (SQER)**.

The PFLASH read cycles are inherited from mode prior to entering **Error Mode**:

- **Operation Mode:** DMU\_HF\_PWAIT.RFLASH and DMU\_HF\_PWAIT.RECC
- **Cranking Mode:** DMU\_HF\_PWAIT.CFLASH and DMU\_HF\_PWAIT.CECC

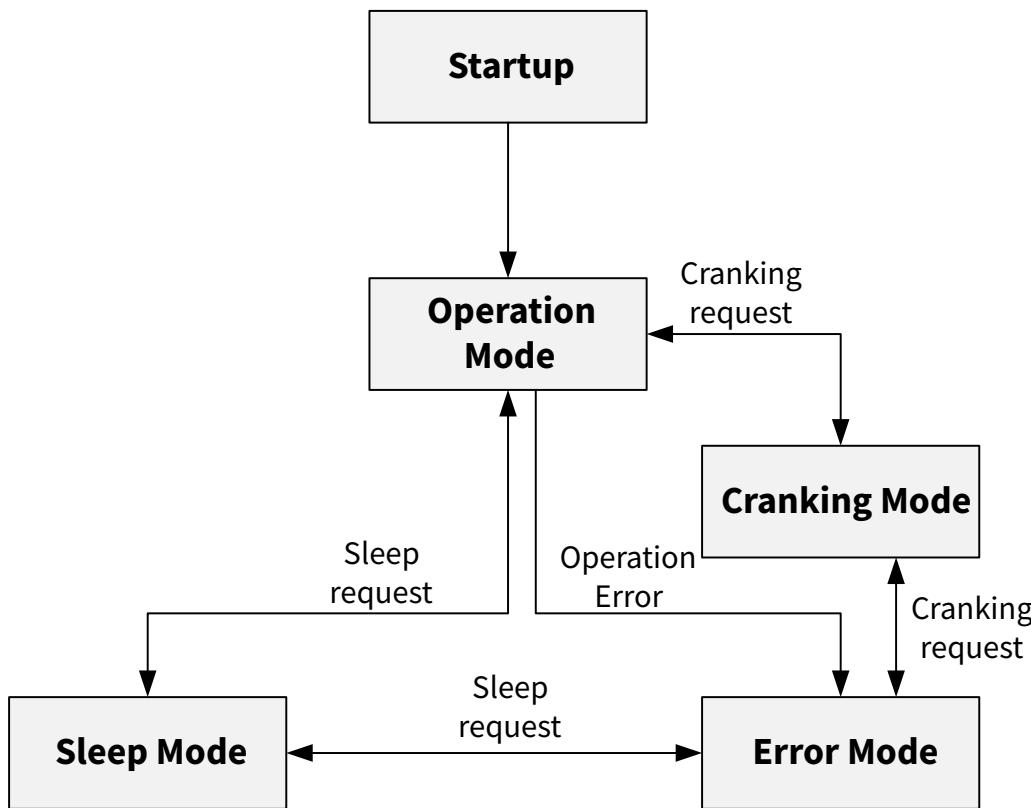
**Cranking Mode, Demand Mode** and **Dynamic Idle Mode** are inherited from the mode prior to entering **Error Mode** and can also be programmed by software while in **Error Mode**.

A system reset must be applied to exit **Error Mode**.

A sleep request is not acted upon by the DMU.

### 6.5.2.6.3 Power modes

For power reduction, DMU can be programmed to be in **Sleep Mode** or **Cranking Mode**.

**Figure 63 Modes**

### 6.5.2.7 Internal Connections

#### 6.5.2.7.1 Clocks

**Table 148 Clocks of Flash Interfaces**

Clock Name	Description of Use in Module
$f_{SRI}$	SRI clock (used in PFI and DMU)
$f_{BROM}$	BROM clock
$f_{FSI}$	FSI clock (used in FSI, PFI and DMU)
$f_{FSI2}$	FSI PFlash read clock (used in PFRWB), must be greater than or equal to $f_{SRI}$ for correct PFLASH operations

#### 6.5.2.7.2 Interrupts and Service Requests

The DMU peripheral generates an interrupt service request.

**Table 149 Interrupt /Service Request Sources of DMU**

Service Request Name	Description of Interrupt Source
DMUHOST	<p>Host Command Interface interrupt service request supports</p> <ul style="list-style-type: none"> <li>• Indication of end of busy</li> <li>• Indication of an operation error</li> <li>• Indication of a protection error</li> <li>• Indication of a command sequence error</li> <li>• Indication of a program or erase verify error</li> </ul>
DMUHSM	<p>HSM Command Interface interrupt service request supports (routed to HSM)</p> <ul style="list-style-type: none"> <li>• Indication of end of busy</li> <li>• Indication of an operation error</li> <li>• Indication of a command sequence error</li> <li>• Indication of a program or erase verify error</li> </ul> <p><i>Note:</i> <i>HSM command interface does not generate an “end of busy” interrupt for the falling edge of BUSY after startup.</i></p>
DMUFSI	FSI interrupt enabled for test purposes.

### 6.5.2.7.3 Cross Triggers

The DMU peripheral has no cross triggers.

### 6.5.2.8 Power Modes

Power consumption may be controlled by the following means:

#### 6.5.2.8.1 Flash Prefetch Buffers

Individual flash prefetch buffers are disabled by programming the respective CPUX\_FLASHCON0 bit field to the reserved Master Tag Identifier value. If prefetch accesses are disabled by changing mode or by disabling individual prefetch buffers then it does not prevent the delivery of existing prefetch data.

#### 6.5.2.8.2 Demand Mode

When in Demand Mode, PFLASH prefetch accesses to the local PFLASH bank by the flash prefetch buffers are disabled. Demand Mode can be requested by writing  $1_B$  to DMU\_HF\_PCONTROL.DEMAND. Demand Mode is also sub-mode of **Cranking Mode** and is entered when DMU\_HF\_CCONTROL.CRANKING =  $11_B$ .

#### 6.5.2.8.3 Dynamic Idle Mode

When there is no flash access the wordline decoder is switched off.

- DFLASH: dynamic idle is automatically enabled by hardware when no read access is performed.
- PFLASH: dynamic idle is enabled by software.

#### 6.5.2.8.4 Sleep Mode

Switching the DMU to Sleep Mode is requested by software. The 1.2V and 3.3V supply voltages to all flash modules (banks) are powered up. The regulators and charge pumps are powered down. The clock is switched off.

The sleep request can have two sources:

- Global sleep mode requested by SCU. Only executed by DMU when DMU\_HF\_PCONTROL.ESLDIS =  $00_B$
- Writing  $11_B$  to DMU\_HF\_PCONTROL.SLEEP

After receiving a sleep request, the Flash starts to ramp down and becomes idle, i.e. none of the banks are in command mode and no reads are executed anymore. An ongoing read burst is finished completely. During ramp down to sleep mode, all the BUSY status bits in the DMU registers are set.

The sleep mode is indicated in DMU\_HF\_PSTATUS.SLEEP. All the BUSY status bits in the DMU registers remain set.

**Note:** Requesting sleep mode does not disable automatically an enabled end-of-busy interrupt. When requesting sleep mode during an ongoing Flash operation with enabled end-of-busy interrupt, the operation finishes, the interrupt is issued and then the Flash enters sleep mode. However, this end-of-busy interrupt will wake-up the CPU again.

#### 6.5.2.8.5 Cranking Mode

The 3.3V supply voltage to all PFLASH and DFLASH banks is reduced, the 5V supply is not reliable. The read access time to all flash modules is increased. Flash prefetch buffer accesses are disabled. **Cranking Mode** is enabled by software writing DMU\_HF\_CCONTROL.CRANKING.

Control of the power sub-modes is:

- **Demand Mode:** flash prefetch buffer accesses are disabled by hardware.
- **Dynamic Idle Mode:** may be configured by software.

Only the **DMU Commands** and FSI ROM Commands (see FSI chapter for details) are supported. The remaining commands fail with **Sequence Error (SQER)**.

Before entering **Cranking Mode** software must suspend all on-going flash operations. On exiting **Cranking Mode** and after entering **Operation Mode** suspended flash operations may be resumed with **Resume NVM Operation**.

The PFLASH read cycles are calculated from DMU\_HF\_PWAIT.CFLASH and DMU\_HF\_PWAIT.CECC

### 6.5.2.8.6 Standby Mode

Device wide mode. The 1.2V supply voltage to all flash modules (banks) is powered down. Standby mode shall be entered only if the DMU is in sleep or read mode (i.e., none of the flash banks are in command mode, and performing any flash operations).

### 6.5.2.9 Boot ROM (BROM)

#### 6.5.2.9.1 Read Accesses

Both address ranges of the BROM (i.e. the cached and the non-cached) support SRI single and burst transfers.

#### 6.5.2.9.2 Data Integrity

The data in the BROM is ECC protected.

### 6.5.3 Registers

**Table 150 Flash Interface SFR Naming Convention**

Module Name	Group Name	Description
PMU	N/A	PMU SFR - PMU does not exist as a hardware module or chapter (replaced with DMU, PFI and separate NVM modules and chapters), but the name is retained for the Flash ID, BootROM and Tuning Protection registers for backward compatibility of software to TC39x-A Step. • BROM Control
PFI0		Flash Flash Interface 0
PFI1		Flash Flash Interface 1 (if PFI1 exists).
PFI2		Flash Flash Interface 2 (if PFI2 exists).
PFI3		Flash Flash Interface 3 (if PFI3 exists).
PFI4		Flash Flash Interface 4 (if PFI4 exists).
PFI5		Flash Flash Interface 5 (if PFI5 exists).
DMU	HF	Host Command Interface • Host Command Sequence Interpreter
DMU	HP	Host Protection Configuration
DMU	SF	HSM Command Interface • HSM Command Sequence Interpreter
DMU	SP	HSM Protection Configuration
FSI	N/A	FSI SFR

### 6.5.3.1 Flash ID and BootROM Registers (PMU)

**Table 151 Register Address Space - PMU**

Module	Base Address	End Address	Note
PMU	F8038000 <sub>H</sub>	F803FFFF <sub>H</sub>	sri slave interface

**Table 152 Register Overview - PMU (ascending Offset Address)**

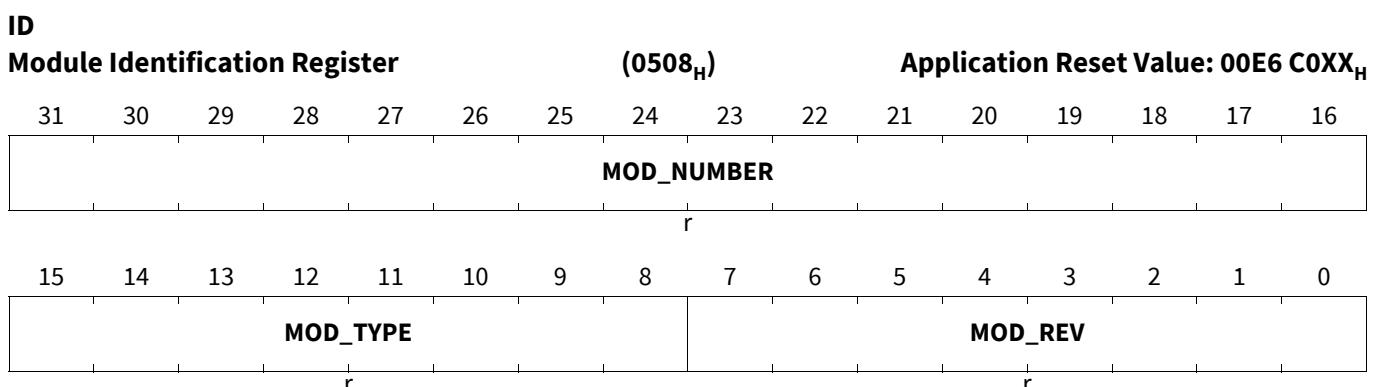
Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
ID	Module Identification Register	0508 <sub>H</sub>	U,SV	BE	Application Reset	58

#### 6.5.3.1.1 PMU Identification

The PMU\_ID register identifies the DMU and its version.

##### Module Identification Register

This module identification register identifies the DMU module



Field	Bits	Type	Description
<b>MOD_REV</b>	7:0	r	<b>Module Revision Number</b> MOD_REV defines the module revision number.
<b>MOD_TYPE</b>	15:8	r	<b>Module Type</b> This bit field is C0 <sub>H</sub> . It defines the module as a 32-bit module.
<b>MOD_NUMBE R</b>	31:16	r	<b>Module Number Value</b> This bit field defines the module identification number.

### 6.5.3.2 DMU Registers

**Table 153 Register Address Space - DMU**

Module	Base Address	End Address	Note
(DMU)	8FFF0000 <sub>H</sub>	8FFFFFFF <sub>H</sub>	Boot ROM (BROM)
	AF000000 <sub>H</sub>	AF0FFFFF <sub>H</sub>	Data Flash 0 EEPROM (DF0) and Host Command Sequence Interpreter
	AFC00000 <sub>H</sub>	AFC1FFFF <sub>H</sub>	Data Flash 1 EEPROM (DF1) and HSM Command Sequence Interpreter
	AFFF0000 <sub>H</sub>	AFFFFFFF <sub>H</sub>	Boot ROM (BROM)
DMU	F8040000 <sub>H</sub>	F807FFFF <sub>H</sub>	SRI slave interface - Register Address Space
(DMU)	FFC00000 <sub>H</sub>	FFC1FFFF <sub>H</sub>	Data Flash 1 EEPROM (DF1) and HSM Command Sequence Interpreter

**Table 154 Register Overview - DMU (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
HF_ID	Module Identification Register	0000008 <sub>H</sub>	U,SV	BE	Application Reset	63
HF_STATUS	Flash Status Register	0000010 <sub>H</sub>	U,SV	BE	Application Reset	63
HF_CONTROL	Flash Control Register	0000014 <sub>H</sub>	U,SV	P,SV,E	Application Reset	65
HF_OPERATION	Flash Operation Register	0000018 <sub>H</sub>	U,SV	BE	System Reset	67
HF_PROTECT	Flash Protection Status Register	000001C <sub>H</sub>	U,SV	BE	Application Reset	68
HF_CONFIRM0	Flash Confirm Status Register 0	0000020 <sub>H</sub>	U,SV	BE	Application Reset	70
HF_CONFIRM1	Flash Confirm Status Register 1	0000024 <sub>H</sub>	U,SV	BE	Application Reset	73
HF_CONFIRM2	Flash Confirm Status Register 2	0000028 <sub>H</sub>	U,SV	BE	Application Reset	75
HF_EER	Enable Error Interrupt Control Register	0000030 <sub>H</sub>	U,SV	P,SV	Application Reset	78
HF_ERRSR	Error Status Register	0000034 <sub>H</sub>	U,SV	BE	Application Reset	79
HF_CLRE	Clear Error Register	0000038 <sub>H</sub>	U,SV	P,SV	Application Reset	81
HF_ECCR	DF0 ECC Read Register	0000040 <sub>H</sub>	U,SV	BE	Application Reset	82
HF_ECCS	DF0 ECC Status Register	0000044 <sub>H</sub>	U,SV	BE	Application Reset	82

**Table 154 Register Overview - DMU (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
HF_ECCC	DF0 ECC Control Register	0000048 <sub>H</sub>	U,SV	P,SV,E	Application Reset	<a href="#">85</a>
HF_ECCW	DF0 ECC Write Register	000004C <sub>H</sub>	U,SV	P,SV,E	Application Reset	<a href="#">86</a>
HF_CCONTROL	Cranking Control Register	0000050 <sub>H</sub>	U,SV	P,SV	System Reset	<a href="#">88</a>
HF_PSTATUS	Power Status Register	0000060 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">89</a>
HF_PCONTROL	Power Control Register	0000064 <sub>H</sub>	U,SV	P,SV	Application Reset	<a href="#">89</a>
HF_PWAIT	PFLASH Wait Cycle Register	0000068 <sub>H</sub>	U,SV	P,SV,E	System Reset	<a href="#">90</a>
HF_DWAIT	DFLASH Wait Cycle Register	000006C <sub>H</sub>	U,SV	P,SV,E	System Reset	<a href="#">92</a>
HF_PROCONUSR	DF0 User Mode Control	0000074 <sub>H</sub>	U,SV	BE	See page <a href="#">87</a>	<a href="#">87</a>
HF_PROCONPF	PFLASH Protection Configuration	0000080 <sub>H</sub>	U,SV	BE	See page <a href="#">92</a>	<a href="#">92</a>
HF_PROCONTP	Tuning Protection Configuration	0000084 <sub>H</sub>	U,SV	BE	See page <a href="#">93</a>	<a href="#">93</a>
HF_PROCONDIF	DFLASH Protection Configuration	0000088 <sub>H</sub>	U,SV	BE	See page <a href="#">94</a>	<a href="#">94</a>
HF_PROCONRAM	RAM Configuration	000008C <sub>H</sub>	U,SV	BE	See page <a href="#">96</a>	<a href="#">96</a>
HF_PROCONDBG	Debug Interface Protection Configuration	0000090 <sub>H</sub>	U,SV	BE	See page <a href="#">97</a>	<a href="#">97</a>
HF_SUSPEND	Suspend Control Register	00000F0 <sub>H</sub>	U,SV	P,U,SV	Application Reset	<a href="#">99</a>
HF_MARGIN	Margin Control Register	00000F4 <sub>H</sub>	U,SV	P,U,SV	Application Reset	<a href="#">100</a>
HF_ACCEN1	Access Enable Register 1	00000F8 <sub>H</sub>	U,SV	SV,SE	Application Reset	<a href="#">102</a>
HF_ACCENO	Access Enable Register 0	00000FC <sub>H</sub>	U,SV	SV,SE	Application Reset	<a href="#">101</a>
HP_PROCONPi0	PFLASH Bank i Protection Configuration 0	0010000 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page <a href="#">103</a>	<a href="#">103</a>
HP_PROCONPi1	PFLASH Bank i Protection Configuration 1	0010004 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page <a href="#">103</a>	<a href="#">103</a>
HP_PROCONPi2	PFLASH Bank i Protection Configuration 2	0010008 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page <a href="#">104</a>	<a href="#">104</a>

**Table 154 Register Overview - DMU (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
HP_PROCONPi3	PFLASH Bank i Protection Configuration 3	001000C <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 104	104
HP_PROCONPi4	PFLASH Bank i Protection Configuration 4	0010010 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 105	105
HP_PROCONPi5	PFLASH Bank i Protection Configuration 5	0010014 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 106	106
HP_PROCONOTP_i0	PFLASH Bank i OTP Protection Configuration 0	0010040 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 106	106
HP_PROCONOTP_i1	PFLASH Bank i OTP Protection Configuration 1	0010044 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 107	107
HP_PROCONOTP_i2	PFLASH Bank i OTP Protection Configuration 2	0010048 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 108	108
HP_PROCONOTP_i3	PFLASH Bank i OTP Protection Configuration 3	001004C <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 108	108
HP_PROCONOTP_i4	PFLASH Bank i OTP Protection Configuration 4	0010050 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 109	109
HP_PROCONOTP_i5	PFLASH Bank i OTP Protection Configuration 5	0010054 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 110	110
HP_PROCONWO_Pi0	PFLASH Bank i WOP Configuration 0	0010080 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 110	110
HP_PROCONWO_Pi1	PFLASH Bank i WOP Configuration 1	0010084 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 111	111
HP_PROCONWO_Pi2	PFLASH Bank i WOP Configuration 2	0010088 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 112	112
HP_PROCONWO_Pi3	PFLASH Bank i WOP Configuration 3	001008C <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 112	112
HP_PROCONWO_Pi4	PFLASH Bank i WOP Configuration 4	0010090 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 113	113
HP_PROCONWO_Pi5	PFLASH Bank i WOP Configuration 5	0010094 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 114	114
HP_ECPRIOf0	PFLASH Bank i Erase Counter Priority configuration 0	00100A0 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 114	114
HP_ECPRIOf1	PFLASH Bank i Erase Counter Priority Configuration 1	00100A4 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 115	115
HP_ECPRIOf2	PFLASH Bank i Erase Counter Priority Configuration 2	00100A8 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 116	116

**Table 154 Register Overview - DMU (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
HP_ECPRI0i3	PFLASH Bank i Erase Counter Priority Configuration 3	00100AC <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 116	116
HP_ECPRI0i4	PFLASH Bank i Erase Counter Priority Configuration 4	00100B0 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 117	117
HP_ECPRI0i5	PFLASH Bank i Erase Counter Priority Configuration 5	00100B4 <sub>H</sub> +i*100 <sub>H</sub>	U,SV	BE	See page 117	117
SF_STATUS	HSM Flash Status Register	0020010 <sub>H</sub>	H	BE	Application Reset	118
SF_CONTROL	HSM Flash Configuration Register	0020014 <sub>H</sub>	H	H	Application Reset	119
SF_OPERATION	HSM Flash Operation Register	0020018 <sub>H</sub>	H	BE	System Reset	120
SF_EER	HSM Enable Error Interrupt Control Register	0020030 <sub>H</sub>	H	H	Application Reset	121
SF_ERRSR	HSM Error Status Register	0020034 <sub>H</sub>	H	BE	Application Reset	122
SF_CLRE	HSM Clear Error Register	0020038 <sub>H</sub>	H	H	Application Reset	123
SF_ECCR	HSM DF1 ECC Read Register	0020040 <sub>H</sub>	H	BE	Application Reset	123
SF_ECCS	HSM DF1 ECC Status Register	0020044 <sub>H</sub>	H	BE	Application Reset	124
SF_ECCC	HSM DF1 ECC Control Register	0020048 <sub>H</sub>	H	H	Application Reset	127
SF_ECCW	HSM DF1 ECC Write Register	002004C <sub>H</sub>	H	H	Application Reset	128
SF_PROCONUSR	HSM DF1 User Mode Control	0020074 <sub>H</sub>	U,SV	BE	See page 129	129
SF_SUSPEND	HSM Suspend Control Register	00200E8 <sub>H</sub>	H	H	Application Reset	129
SF_MARGIN	HSM DF1 Margin Control Register	00200EC <sub>H</sub>	H	H	Application Reset	130
SP_PROCONHSM CFG	HSM Protection Configuration	0030000 <sub>H</sub>	U,SV	BE	See page 131	131
SP_PROCONHSM CBS	HSM Code Boot Sector	0030004 <sub>H</sub>	U,SV	BE	See page 133	133
SP_PROCONHSM CX0	HSM Code Exclusive Protection Configuration	0030008 <sub>H</sub>	U,SV	BE	See page 135	135

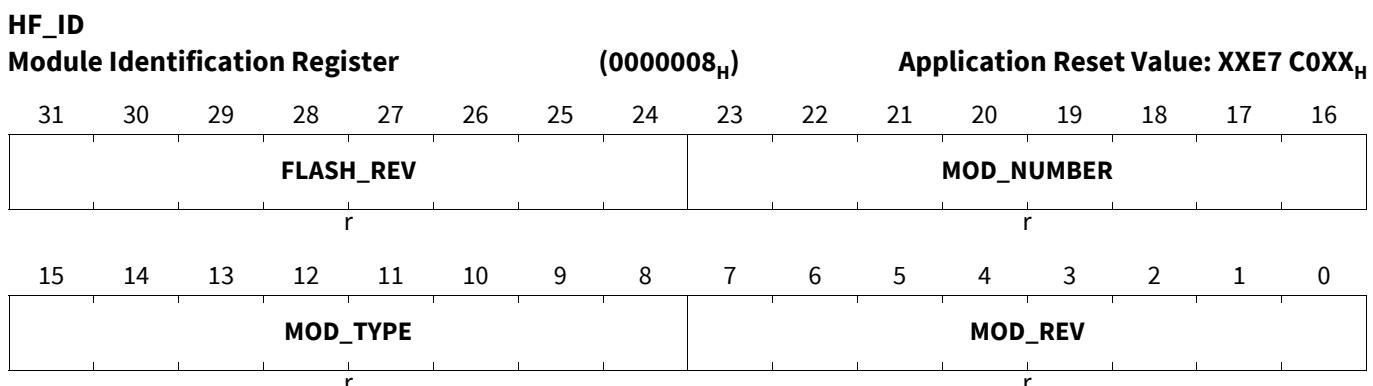
**Table 154 Register Overview - DMU (ascending Offset Address) (cont'd)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
SP_PROCONHSM_CX1	HSM Code Exclusive Protection Configuration	003000C <sub>H</sub>	U,SV	BE	See page 136	136
SP_PROCONHSM_COTP0	HSM Code OTP Protection Configuration	0030010 <sub>H</sub>	U,SV	BE	See page 136	136
SP_PROCONHSM_COTP1	HSM Code OTP Protection Configuration	0030014 <sub>H</sub>	U,SV	BE	See page 137	137
SP_PROCONHSM	HSM Interface Protection Configuration	0030040 <sub>H</sub>	U,SV	BE	See page 138	138

### 6.5.3.2.1 DMU Identification

#### Module Identification Register

The module identification register identifies the Flash module.



Field	Bits	Type	Description
<b>MOD_REV</b>	7:0	r	<b>Module Revision Number</b> This bit field defines the module revision number reflecting major changes in the module.
<b>MOD_TYPE</b>	15:8	r	<b>Module Type</b> The bit field is set to C0 <sub>H</sub> which defines the module as a 32-bit module.
<b>MOD_NUMBE R</b>	23:16	r	<b>Module Number Value</b> This bit field defines the module identification number.
<b>FLASH_REV</b>	31:24	r	<b>Flash Revision</b> This bit field defines the flash revision number.

### 6.5.3.2.2 Host Command Interface

#### Flash Status Register

The Flash Status Register reflects the status of the Flash Banks after reset.

**Note:** The DxBUSY and PxBUSY flags cannot be cleared with the “Clear Status” command or with the “Reset to Read” command. These flags are controlled by HW.

**Note:** After every reset, the busy bits are set while the Flash module is busy with startup (until the operation mode is entered). Also the protection installation bits are always set until end of startup.

HF STATUS

## **Flash Status Register**

(0000010<sub>H</sub>)

## **Application Reset Value: 0000 00FF**

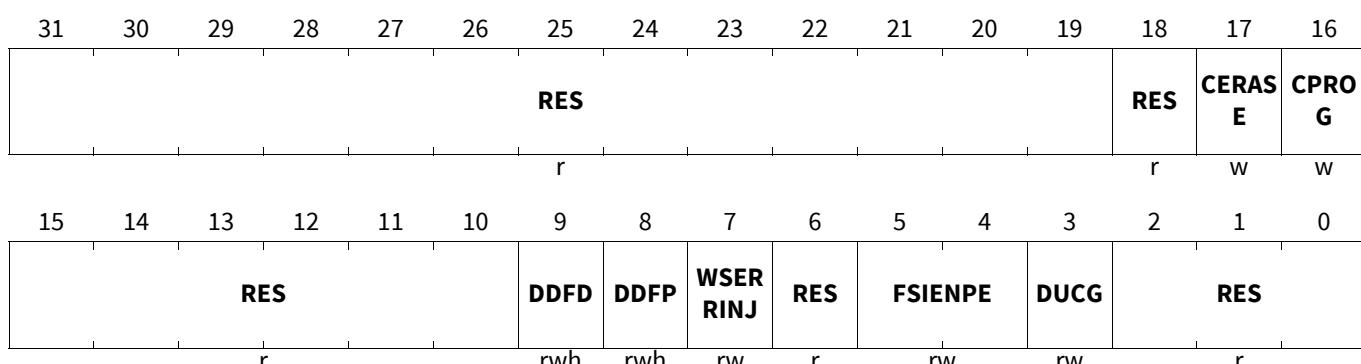
Field	Bits	Type	Description
<b>D0BUSY</b>	0	rh	<p><b>Data Flash Bank 0 Busy</b>            HW-controlled status flag.            Indication of busy state of DFLASH bank 0 because of active execution of an operation; DF0 busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state the DF0 does not allow read access.</p> <p><math>0_B</math> DF0 ready, not busy; DF0 in operation mode.  <math>1_B</math> DF0 busy; DF0 not in operation mode.</p>
<b>D1BUSY</b>	1	rh	<p><b>Data Flash Bank 1 Busy</b>            HW-controlled status flag.            Indication of busy state of DFLASH bank 1 because of active execution of an operation; DF1 busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state the DF1 does not allow read access.</p> <p>Bit is not set for program/erase operations initiated by the HSM interface.</p> <p><math>0_B</math> DF1 ready, not busy; DF1 in operation mode.  <math>1_B</math> DF1 busy; DF1 not in operation mode.</p>
<b>PxBUSY (x=0-5)</b>	x+2	rh	<p><b>Program Flash PFxBUSY</b>            HW-controlled status flag.            Indication of busy state of PFx because of active execution of an operation; PFx busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state the PFx does not allow read access.</p> <p><math>0_B</math> PFx ready, not busy; PFx in operation mode.  <math>1_B</math> PFx busy; PFx not in operation mode.</p>

Field	Bits	Type	Description
<b>RES</b>	15:8, 23, 31:26	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>RES</b>	16, 17, 18, 19, 22, 25:24	rX	<b>Reserved</b> Undefined.
<b>DFPAGE</b>	20	rh	<b>Data Flash in Page Mode</b> HW-controlled status flag. Set with Enter Page Mode for DFLASH, cleared with Write Page command. This bit is not set by “Enter Page Mode” initiated by the HSM interface.  Note: <i>Read accesses are allowed while in page mode.</i>  0 <sub>B</sub> Data Flash not in page mode 1 <sub>B</sub> Data Flash in page mode
<b>PFPAGE</b>	21	rh	<b>Program Flash in Page Mode</b> HW-controlled status flag. Set with Enter Page Mode for Flash, cleared with Write Page command This bit is not set by “Enter Page Mode” initiated by the HSM interface.  Note: <i>Read accesses are allowed while in page mode.</i>  0 <sub>B</sub> Flash not in page mode. 1 <sub>B</sub> Flash in page mode.

## Flash Control Register

### HF\_CONTROL

#### Flash Control Register

(00000014<sub>H</sub>)Application Reset Value: 0000 0320<sub>H</sub>

Field	Bits	Type	Description
<b>RES</b>	2:0, 6, 15:10, 18, 31:19	r	<b>Reserved</b>
<b>DUCG</b>	3	rw	<b>DFLASH User Command Granularity</b> Granularity configuration of User commands for DFLASH0 $0_B$ Wordline granularity. $1_B$ Page granularity.
<b>FSIENPE</b>	5:4	rw	<b>Enable Program/Erase</b> The field prevents any Flash program or erase directly in the FSI. It is set to “Enabled” by the SSW upon finishing the startup.  <i>Note:</i> Once in the “enabled” state this field cannot be changed.  $00_B$ <b>Disabled</b> , Program/Erase are disabled in the FSI. $01_B$ <b>Enabled</b> , Program/Erase are enabled in the FSI. $10_B$ <b>Disabled</b> , Program/Erase are disabled in the FSI. $11_B$ <b>Disabled</b> , Program/Erase are disabled in the FSI.
<b>WSERRINJ</b>	7	rw	<b>PFlash Wait State ECC error injection</b> Error injection into the ECC logic protecting PFlash Wait states. $0_B$ No error injection requested $1_B$ An error is injected into the ECC protecting PFlash wait states
<b>DDFP</b>	8	rwh	<b>Disable Read from PFLASH</b> This bit enables/disables the read access to PFLASH. <b>DMU ramp up and SSW start up</b> This bit is automatically set with reset and is cleared during startup, if no Read Protection installed, and during startup SSW in case of internal start out of Flash. <b>User Software: Clearing PFLASH Read Protection</b> Once set, <b>HF_CONTROL.DDFP</b> can only be cleared when <b>HF_PROTECT.PRODISP</b> or <b>HF_PROTECT.PRODISP0-5</b> or not <b>HF_PROCONPF.RPRO</b> <b>User Software: Setting PFLASH Read Protection</b> Software must write <b>HF_CONTROL.DDFP</b> to $1_B$ $0_B$ Read access from the PFLASH memory area is allowed. $1_B$ Read access from the PFLASH memory area is not allowed.

Field	Bits	Type	Description
<b>DDFD</b>	9	rwh	<p><b>Disable Data Fetch from DF0_EEPROM</b>  This bit enables/disables the data fetch from DF0_EEPROM.</p> <p><b>DMU ramp up and SSW start up</b>  This bit is automatically set with reset and is cleared during startup, if no Read Protection installed, and during startup (BROM SW) in case of internal start out of Flash.</p> <p><b>User Software: Clearing DF0_EEPROM Read Protection</b>  Once set, <b>HF_CONTROL.DDFD</b> can only be cleared when <b>HF_PROTECT.PRODISD</b> or not <b>HF_PROCONDF.RPRO</b></p> <p><b>User Software: Setting DF0_EEPROM Read Protection</b>  Software must write <b>HF_CONTROL.DDFD</b> to 1<sub>B</sub></p> <p>0<sub>B</sub> Read access from the DF0_EEPROM memory area is allowed.  1<sub>B</sub> Read access from the DF0_EEPROM memory area is not allowed.</p>
<b>CPROG</b>	16	w	<p><b>Clear Programming State</b>  0<sub>B</sub> No action.  1<sub>B</sub> Clear the Programming State Flag DMU_HF_OPERATION.PROG</p>
<b>CERASE</b>	17	w	<p><b>Clear Erase State</b>  0<sub>B</sub> No action.  1<sub>B</sub> Clear the Erase State Flag DMU_HF_OPERATION.ERASE</p>

### PFLASH and DFLASH Read Protection

After DMU ramp up the SSW will set the following read protection values:

- Start not in internal Flash:
  - **HF\_CONTROL.DDFP** is set to **HF\_PROCONPF.RPRO**
  - **HF\_CONTROL.DDFD** is set to **HF\_PROCONDF.RPRO**
- Start in internal Flash:
  - **HF\_CONTROL.DDFP** is cleared to 0<sub>B</sub>
  - **HF\_CONTROL.DDFD** is cleared to 0<sub>B</sub>

Before disabling read access by setting **HF\_CONTROL.DDFP** or **HF\_CONTROL.DDFD** all pending read accesses to the affected Flash ranges should have finished.

### Flash Operation Register

The Flash Operation Register reflects the overall status of the Flash after reset.

#### HF\_OPERATION

##### Flash Operation Register

(00000018<sub>H</sub>)

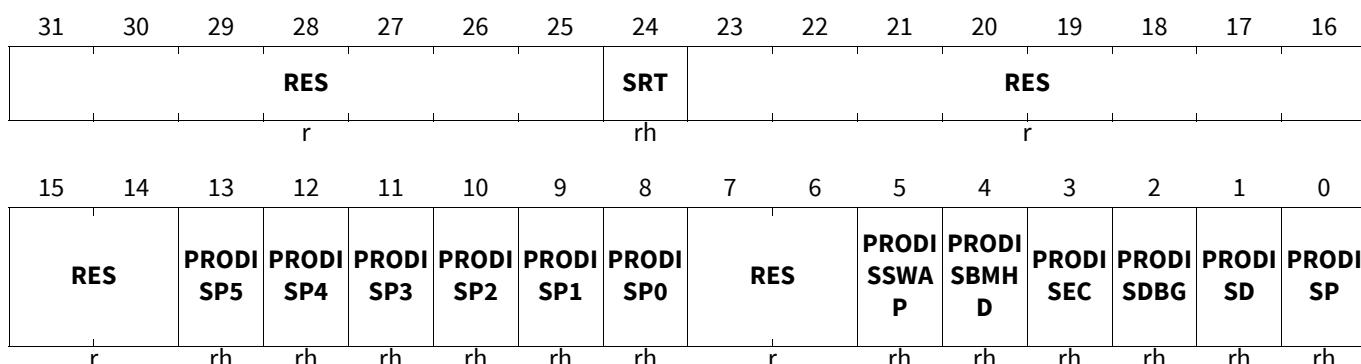
System Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				RES			RES				RES		ERASE	PROG	

Field	Bits	Type	Description
<b>PROG</b>	0	rh	<p><b>Programming State</b></p> <p>HW-controlled status flag.</p> <p>Set with last cycle of Write Page/Burst and Replace Logical Sector command sequences.</p> <p>If one BUSY flag is coincidentally set, PROG indicates the type of busy state.</p> <p>If OPER is coincidentally set, PROG indicates the type of erroneous operation. Otherwise, PROG indicates, that operation is still requested or finished.</p> <p>May be also cleared by writing ‘1’ to <a href="#">HF_CONTROL.CPROG</a>.</p> <p>This bit is not set for by program operations initiated by the HSM interface.</p> <p><i>Note:</i> Cleared with command “Clear Status”.</p> <p>0<sub>B</sub> There is no program operation requested or in progress or just finished. 1<sub>B</sub> Programming operation requested or in action or finished.</p>
<b>ERASE</b>	1	rh	<p><b>Erase State</b></p> <p>HW-controlled status flag.</p> <p>Set with last cycle of Erase/Verify command sequence.</p> <p>Indications are analogous to PROG flag.</p> <p>May be cleared by writing ‘1’ to <a href="#">HF_CONTROL.CERASE</a>.</p> <p>This bit is not set for by erase operations initiated by the HSM interface.</p> <p><i>Note:</i> Cleared with command “Clear Status”.</p> <p>0<sub>B</sub> There is no erase operation requested or in progress or just finished 1<sub>B</sub> Erase/Verify operation requested or in action or finished.</p>
<b>RES</b>	2, 7:3, 10:8, 31:11	r	<b>Reserved</b>

### Flash Protection Status Register

This register reports the state of the Flash protection and contains protection relevant control fields.

**HF\_PROTECT****Flash Protection Status Register**(0000001C<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>PRODISP</b>	0	rh	<p><b>PFLASH Protection Disabled</b>            The protection configured by UCB_PFLASH_ORIG and UCB_PFLASH_COPY was successfully disabled by supplying the correct password to “Disable Protection”.</p> <p><i>Note:</i> Cleared with command “Resume Protection”.</p>
<b>PRODISD</b>	1	rh	<p><b>DFLASH Protection Disabled</b>            The protection configured by UCB_DFLASH_ORIG and UCB_DFLASH_COPY was successfully disabled by supplying the correct password to “Disable Protection”.</p> <p><i>Note:</i> Cleared with command “Resume Protection”.</p>
<b>PRODISDBG</b>	2	rh	<p><b>Debug Interface Password Protection Disabled</b>            The password configured by UCB_DBG_ORIG and UCB_DBG_COPY was correctly received with “Disable Protection”.            When DMU_SP_PROCONHSMCFG.DESTDBG is “destructive” then only the SSW can disable this protection.</p> <p><i>Note:</i> Cleared with command “Resume Protection”.</p>
<b>PRODISSEC</b>	3	rh	<p><b>Erase Counter Priority Protection Disabled</b>            The protection configured by UCB_ECPPIO_ORIG and UCB_ECPPIO_COPY was successfully disabled by supplying the correct password to “Disable Protection”.</p> <p><i>Note:</i> Cleared with command “Resume Protection”.</p>
<b>PRODISBMHD</b>	4	rh	<p><b>BMHD Protection Disabled</b>            The protection configured by UCB_BMHD0_ORIG and UCB_BMHD0_COPY was successfully disabled by supplying the correct password to “Disable Protection”.</p> <p><i>Note:</i> Cleared with command “Resume Protection”.</p>

Field	Bits	Type	Description
<b>PRODISSWAP</b>	5	rh	<b>UCB_SWAP protection Disabled</b> The protection configured by UCB_SWAP_ORIG and UCB_SWAP_COPY was successfully disabled by supplying the correct password to "Disable Protection".  Note: Cleared with command "Resume Protection".
<b>RES</b>	7:6, 23:14, 31:25	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>PRODISPx (x=0-5)</b>	x+8	rh	<b>Program Flash Protection Disable PRODISPx</b> The protection configured for PFx by UCB_PFLASH_ORIG and UCB_PFLASH_COPY was successfully disabled by supplying the correct password to "Disable Protection".  Note: Cleared with command "Resume Protection".
<b>SRT</b>	24	rh	<b>Secure Retest Password Protection Disabled</b>  Note: Cleared with command "Resume Protection".  0 <sub>B</sub> Secure Retest protection is not disabled. 1 <sub>B</sub> Secure Retest protection is disabled.

### Flash Confirm Status Register 0

This register reports the state of the UCB confirmation codes.

Note: After reset and execution of BROM startup SW, the PROIN fields are set depending on the content of their assigned UCB sector

#### HF\_CONFIRM0

##### Flash Confirm Status Register 0

(0000020<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>PROINSRT</b>	<b>RES</b>				<b>PROINREDSE</b>	<b>PROINBMHD3</b>	<b>PROINBMHD2</b>	<b>PROINBMHD1</b>	<b>PROINBMHD0</b>						
rh		r			rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>PROINHSMCF G</b>	<b>PROINTEST</b>	<b>PROINUSER</b>	<b>PROINSSW</b>	<b>PROINBMHD3</b>	<b>PROINBMHD2</b>	<b>PROINBMHD1</b>	<b>PROINBMHD0</b>								
rh		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
<b>PROINBMHD0_O</b>	1:0	rh	<b>UCB_BMHD0_ORIG Confirmation</b> This bit reflects the confirmed state of UCB_BMHD0_ORIG. $00_B$ UCB_BMHD0_ORIG state is UNREAD. $01_B$ UCB_BMHD0_ORIG state is UNLOCKED. $10_B$ UCB_BMHD0_ORIG state is CONFIRMED. $11_B$ UCB_BMHD0_ORIG state is ERRORRED.
<b>PROINBMHD1_O</b>	3:2	rh	<b>UCB_BMHD1_ORIG Confirmation</b> This bit reflects the confirmed state of UCB_BMHD1_ORIG. $00_B$ UCB_BMHD1_ORIG state is UNREAD. $01_B$ UCB_BMHD1_ORIG state is UNLOCKED. $10_B$ UCB_BMHD1_ORIG state is CONFIRMED. $11_B$ UCB_BMHD1_ORIG state is ERRORRED.
<b>PROINBMHD2_O</b>	5:4	rh	<b>UCB_BMHD2_ORIG Confirmation</b> This bit reflects the confirmed state of UCB_BMHD2_ORIG. $00_B$ UCB_BMHD2_ORIG state is UNREAD. $01_B$ UCB_BMHD2_ORIG state is UNLOCKED. $10_B$ UCB_BMHD2_ORIG state is CONFIRMED. $11_B$ UCB_BMHD2_ORIG state is ERRORRED.
<b>PROINBMHD3_O</b>	7:6	rh	<b>UCB_BMHD3_ORIG Confirmation</b> This bit reflects the confirmed state of UCB_BMHD3_ORIG. $00_B$ UCB_BMHD3_ORIG state is UNREAD. $01_B$ UCB_BMHD3_ORIG state is UNLOCKED. $10_B$ UCB_BMHD3_ORIG state is CONFIRMED. $11_B$ UCB_BMHD3_ORIG state is ERRORRED.
<b>PROINSSW</b>	9:8	rh	<b>UCB_SSW Confirmation</b> This bit reflects the confirmed state of UCB_SSW. $00_B$ UCB_SSW state is UNREAD. $01_B$ UCB_SSW state is UNLOCKED. $10_B$ UCB_SSW state is CONFIRMED. $11_B$ UCB_SSW state is ERRORRED.
<b>PROINUSER</b>	11:10	rh	<b>UCB_USER Confirmation</b> This bit reflects the confirmed state of UCB_USER. $00_B$ UCB_USER state is UNREAD. $01_B$ UCB_USER state is UNLOCKED. $10_B$ UCB_USER state is CONFIRMED. $11_B$ UCB_USER state is ERRORRED.
<b>PROINTEST</b>	13:12	rh	<b>UCB_TEST Confirmation</b> This bit reflects the confirmed state of UCB_TEST. $00_B$ UCB_TEST state is UNREAD. $01_B$ UCB_TEST state is UNLOCKED. $10_B$ UCB_TEST state is CONFIRMED. $11_B$ UCB_TEST state is ERRORRED.

Field	Bits	Type	Description
<b>PROINHSMCF G</b>	15:14	rh	<b>UCB_HSMCFG Confirmation</b> This bit reflects the confirmed state of UCB_HSMCFG. $00_B$ UCB_HSMCFG state is UNREAD. $01_B$ UCB_HSMCFG state is UNLOCKED. $10_B$ UCB_HSMCFG state is CONFIRMED. $11_B$ UCB_HSMCFG state is ERRORRED.
<b>PROINBMHD0 C</b>	17:16	rh	<b>UCB_BMHD0_COPY Confirmation</b> This bit reflects the confirmed state of UCB_BMHD0_COPY. $00_B$ UCB_BMHD0_COPY state is UNREAD. $01_B$ UCB_BMHD0_COPY state is UNLOCKED. $10_B$ UCB_BMHD0_COPY state is CONFIRMED. $11_B$ UCB_BMHD0_COPY state is ERRORRED.
<b>PROINBMHD1 C</b>	19:18	rh	<b>UCB_BMHD1_COPY Confirmation</b> This bit reflects the confirmed state of UCB_BMHD1_COPY. $00_B$ UCB_BMHD1_COPY state is UNREAD. $01_B$ UCB_BMHD1_COPY state is UNLOCKED. $10_B$ UCB_BMHD1_COPY state is CONFIRMED. $11_B$ UCB_BMHD1_COPY state is ERRORRED.
<b>PROINBMHD2 C</b>	21:20	rh	<b>UCB_BMHD2_COPY Confirmation</b> This bit reflects the confirmed state of UCB_BMHD2_COPY. $00_B$ UCB_BMHD2_COPY state is UNREAD. $01_B$ UCB_BMHD2_COPY state is UNLOCKED. $10_B$ UCB_BMHD2_COPY state is CONFIRMED. $11_B$ UCB_BMHD2_COPY state is ERRORRED.
<b>PROINBMHD3 C</b>	23:22	rh	<b>UCB_BMHD3_COPY Confirmation</b> This bit reflects the confirmed state of UCB_BMHD3_COPY. $00_B$ UCB_BMHD3_COPY state is UNREAD. $01_B$ UCB_BMHD3_COPY state is UNLOCKED. $10_B$ UCB_BMHD3_COPY state is CONFIRMED. $11_B$ UCB_BMHD3_COPY state is ERRORRED.
<b>PROINREDSE C</b>	25:24	rh	<b>UCB_REDSEC Confirmation</b> This bit reflects the confirmed state of UCB_REDSEC $00_B$ UCB_REDSEC state is UNREAD. $01_B$ UCB_REDSEC state is UNLOCKED. $10_B$ UCB_REDSEC state is CONFIRMED. $11_B$ UCB_REDSEC state is ERRORRED.
<b>RES</b>	29:26	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>PROINSRT</b>	31:30	rh	<b>UCB_RETEST Confirmation</b> This bit reflects the confirmed state of UCB_RETEST. $00_B$ UCB_RETEST state is UNREAD. $01_B$ UCB_RETEST state is UNLOCKED. $10_B$ UCB_RETEST state is CONFIRMED. $11_B$ UCB_RETEST state is ERRORRED.

**Flash Confirm Status Register 1**

This register reports the state of the UCB confirmation codes.

**Note:** After reset and execution of BROM startup SW, the PROIN fields are set depending on the content of their assigned UCB sectors.

**HF\_CONFIRM1****Flash Confirm Status Register 1**(00000024<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PROINSWAPC	PROINECC	PROINHSMCO TP1C	PROINHSMCO TP0C	PROINHSMC	PROINDBGC	PROINDC	PROINPC								
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROINSWAPO	PROINECO	PROINHSMCO TP10	PROINHSMCO TP00	PROINHSMO	PROINDBGO	PROINDO	PROINPO								
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
PROINPO	1:0	rh	<b>UCB_PFLASH_ORIG Confirmation</b> This bit reflects the confirmed state of UCB_PFLASH_ORIG. 00 <sub>B</sub> UCB_PFLASH_ORIG state is UNREAD. 01 <sub>B</sub> UCB_PFLASH_ORIG state is UNLOCKED. 10 <sub>B</sub> UCB_PFLASH_ORIG state is CONFIRMED. 11 <sub>B</sub> UCB_PFLASH_ORIG state is ERRORED.
PROINDO	3:2	rh	<b>UCB_DFLASH_ORIG Confirmation</b> This bit reflects the confirmed state of UCB_DFLASH_ORIG. 00 <sub>B</sub> UCB_DFLASH_ORIG state is UNREAD. 01 <sub>B</sub> UCB_DFLASH_ORIG state is UNLOCKED. 10 <sub>B</sub> UCB_DFLASH_ORIG state is CONFIRMED. 11 <sub>B</sub> UCB_DFLASH_ORIG state is ERRORED.
PROINDBGO	5:4	rh	<b>UCB_DBG_ORIG Confirmation</b> This bit reflects the confirmed state of UCB_DBG_ORIG. 00 <sub>B</sub> UCB_DBG_COPY state is UNREAD. 01 <sub>B</sub> UCB_DBG_COPY state is UNLOCKED. 10 <sub>B</sub> UCB_DBG_COPY state is CONFIRMED. 11 <sub>B</sub> UCB_DBG_COPY state is ERRORED.
PROINHSMO	7:6	rh	<b>UCB_HSM_ORIG Confirmation</b> This bit reflects the confirmed state of UCB_HSM_ORIG. 00 <sub>B</sub> UCB_HSM_ORIG state is UNREAD. 01 <sub>B</sub> UCB_HSM_ORIG state is UNLOCKED. 10 <sub>B</sub> UCB_HSM_ORIG state is CONFIRMED. 11 <sub>B</sub> UCB_HSM_ORIG state is ERRORED.

Field	Bits	Type	Description
<b>PROINHSMCO TP00</b>	9:8	rh	<b>UCB_HSMCOTP0_ORIG Protection</b> This bit reflects the confirmed state of UCB_HSMCOTP0_ORIG. 00 <sub>B</sub> UCB_HSMCOTP0_ORIG state is UNREAD. 01 <sub>B</sub> UCB_HSMCOTP0_ORIG state is UNLOCKED. 10 <sub>B</sub> UCB_HSMCOTP0_ORIG state is CONFIRMED. 11 <sub>B</sub> UCB_HSMCOTP0_ORIG state is ERRORED.
<b>PROINHSMCO TP10</b>	11:10	rh	<b>UCB_HSMCOTP1_ORIG Protection</b> This bit reflects the confirmed state of UCB_HSMCOTP1_ORIG. 00 <sub>B</sub> UCB_HSMCOTP1_ORIG state is UNREAD. 01 <sub>B</sub> UCB_HSMCOTP1_ORIG state is UNLOCKED. 10 <sub>B</sub> UCB_HSMCOTP1_ORIG state is CONFIRMED. 11 <sub>B</sub> UCB_HSMCOTP1_ORIG state is ERRORED.
<b>PROINECO</b>	13:12	rh	<b>UCB_ECPRIORIG Confirmation</b> This bit reflects the confirmed state of UCB_ECPRIORIG. 00 <sub>B</sub> 00 UCB_ECPRIORIG state is UNREAD. 01 <sub>B</sub> 01 UCB_ECPRIORIG state is UNLOCKED. 10 <sub>B</sub> 10 UCB_ECPRIORIG state is CONFIRMED. 11 <sub>B</sub> 11 UCB_ECPRIORIG state is ERRORED.
<b>PROINSWAPO</b>	15:14	rh	<b>UCB_SWAP_ORIG Confirmation</b> This bit reflects the confirmed state of UCB_SWAP_ORIG. 00 <sub>B</sub> 00 UCB_SWAP_ORIG state is UNREAD. 01 <sub>B</sub> 01 UCB_SWAP_ORIG state is UNLOCKED. 10 <sub>B</sub> 10 UCB_SWAP_ORIG state is CONFIRMED. 11 <sub>B</sub> 11 UCB_SWAP_ORIG state is ERRORED.
<b>PROINPC</b>	17:16	rh	<b>UCB_PFLASH_COPY Confirmation</b> This bit reflects the confirmed state of UCB_PFLASH_COPY. 00 <sub>B</sub> UCB_PFLASH_COPY state is UNREAD. 01 <sub>B</sub> UCB_PFLASH_COPY state is UNLOCKED. 10 <sub>B</sub> UCB_PFLASH_COPY state is CONFIRMED. 11 <sub>B</sub> UCB_PFLASH_COPY state is ERRORED.
<b>PROINDC</b>	19:18	rh	<b>UCB_DFLASH_COPY Confirmation</b> This bit reflects the confirmed state of UCB_DFLASH_COPY. 00 <sub>B</sub> UCB_DFLASH_COPY state is UNREAD. 01 <sub>B</sub> UCB_DFLASH_COPY state is UNLOCKED. 10 <sub>B</sub> UCB_DFLASH_COPY state is CONFIRMED. 11 <sub>B</sub> UCB_DFLASH_COPY state is ERRORED.
<b>PROINDBG</b>	21:20	rh	<b>UCB_DBG_COPY Interface Confirmation</b> This bit reflects the confirmed state of UCB_DBG_COPY. 00 <sub>B</sub> UCB_DBG_COPY state is UNREAD. 01 <sub>B</sub> UCB_DBG_COPY state is UNLOCKED. 10 <sub>B</sub> UCB_DBG_COPY state is CONFIRMED. 11 <sub>B</sub> UCB_DBG_COPY state is ERRORED.

Field	Bits	Type	Description
<b>PROINHSMC</b>	23:22	rh	<b>UCB_HSM_COPY Confirmation</b> This bit reflects the confirmed state of UCB_HSM_COPY. 00 <sub>B</sub> UCB_HSM_COPY state is UNREAD. 01 <sub>B</sub> UCB_HSM_COPY state is UNLOCKED. 10 <sub>B</sub> UCB_HSM_COPY state is CONFIRMED. 11 <sub>B</sub> UCB_HSM_COPY state is ERRORED.
<b>PROINHSMCO TP0C</b>	25:24	rh	<b>UCB_HSMCOTP0_COPY Protection</b> This bit reflects the confirmed state of UCB_HSMCOTP0_COPY. 00 <sub>B</sub> UCB_HSMCOTP0_COPY state is UNREAD. 01 <sub>B</sub> UCB_HSMCOTP0_COPY state is UNLOCKED. 10 <sub>B</sub> UCB_HSMCOTP0_COPY state is CONFIRMED. 11 <sub>B</sub> UCB_HSMCOTP0_COPY state is ERRORED.
<b>PROINHSMCO TP1C</b>	27:26	rh	<b>UCB_HSMCOTP1_COPY Protection</b> This bit reflects the confirmed state of UCB_HSMCOTP1_COPY. 00 <sub>B</sub> UCB_HSMCOTP1_COPY state is UNREAD. 01 <sub>B</sub> UCB_HSMCOTP1_COPY state is UNLOCKED. 10 <sub>B</sub> UCB_HSMCOTP1_COPY state is CONFIRMED. 11 <sub>B</sub> UCB_HSMCOTP1_COPY state is ERRORED.
<b>PROINECC</b>	29:28	rh	<b>UCB_ECPRIOD_COPY Confirmation</b> This bit reflects the confirmed state of UCB_ECPRIOD_COPY 00 <sub>B</sub> 00 UCB_ECPRIOD_COPY state is UNREAD. 01 <sub>B</sub> 01 UCB_ECPRIOD_COPY state is UNLOCKED. 10 <sub>B</sub> 10 UCB_ECPRIOD_COPY state is CONFIRMED. 11 <sub>B</sub> 11 UCB_ECPRIOD_COPY state is ERRORED.
<b>PROINSWAPC</b>	31:30	rh	<b>UCB_SWAP_COPY Confirmation</b> This bit reflects the confirmed state of UCB_SWAP_COPY. 00 <sub>B</sub> 00 UCB_SWAP_COPY state is UNREAD. 01 <sub>B</sub> 01 UCB_SWAP_COPY state is UNLOCKED. 10 <sub>B</sub> 10 UCB_SWAP_COPY state is CONFIRMED. 11 <sub>B</sub> 11 UCB_SWAP_COPY state is ERRORED.

## Flash Confirm Status Register 2

This register reports the state of the UCB confirmation codes.

**Note:** After reset and execution of BROM startup SW, the PROIN fields are set depending on the content of their assigned UCB sector

**HF\_CONFIRM2****Flash Confirm Status Register 2**(00000028<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>PROINOTP7C</b>	<b>PROINOTP6C</b>	<b>PROINOTP5C</b>	<b>PROINOTP4C</b>	<b>PROINOTP3C</b>	<b>PROINOTP2C</b>	<b>PROINOTP1C</b>	<b>PROINOTP0C</b>								
rh	rh	rh	rh	rh	rh	rh	rh	rh							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>PROINOTP7O</b>	<b>PROINOTP6O</b>	<b>PROINOTP5O</b>	<b>PROINOTP4O</b>	<b>PROINOTP3O</b>	<b>PROINOTP2O</b>	<b>PROINOTP1O</b>	<b>PROINOTP0O</b>								
rh	rh	rh	rh	rh	rh	rh	rh	rh							

Field	Bits	Type	Description
<b>PROINOTP0O</b>	1:0	rh	<b>UCB OTP0_ORIG Confirmation</b> This bit reflects the confirmed state of UCB OTP0_ORIG.  00 <sub>B</sub> UCB OTP0_ORIG state is UNREAD. 01 <sub>B</sub> UCB OTP0_ORIG state is UNLOCKED. 10 <sub>B</sub> UCB OTP0_ORIG state is CONFIRMED. 11 <sub>B</sub> UCB OTP0_ORIG state is ERRORRED.
<b>PROINOTP1O</b>	3:2	rh	<b>UCB OTP1_ORIG Confirmation</b> This bit reflects the confirmed state of UCB OTP1_ORIG.  00 <sub>B</sub> UCB OTP1_ORIG state is UNREAD. 01 <sub>B</sub> UCB OTP1_ORIG state is UNLOCKED. 10 <sub>B</sub> UCB OTP1_ORIG state is CONFIRMED. 11 <sub>B</sub> UCB OTP1_ORIG state is ERRORRED.
<b>PROINOTP2O</b>	5:4	rh	<b>UCB OTP2_ORIG Confirmation</b> This bit reflects the confirmed state of UCB OTP2_ORIG.  00 <sub>B</sub> UCB OTP2_ORIG state is UNREAD. 01 <sub>B</sub> UCB OTP2_ORIG state is UNLOCKED. 10 <sub>B</sub> UCB OTP2_ORIG state is CONFIRMED. 11 <sub>B</sub> UCB OTP2_ORIG state is ERRORRED.
<b>PROINOTP3O</b>	7:6	rh	<b>UCB OTP3_ORIG Confirmation</b> This bit reflects the confirmed state of UCB OTP3_ORIG.  00 <sub>B</sub> UCB OTP3_ORIG state is UNREAD. 01 <sub>B</sub> UCB OTP3_ORIG state is UNLOCKED. 10 <sub>B</sub> UCB OTP3_ORIG state is CONFIRMED. 11 <sub>B</sub> UCB OTP3_ORIG state is ERRORRED.
<b>PROINOTP4O</b>	9:8	rh	<b>UCB OTP4_ORIG Confirmation</b> This bit reflects the confirmed state of UCB OTP4_ORIG.  00 <sub>B</sub> UCB OTP4_ORIG state is UNREAD. 01 <sub>B</sub> UCB OTP4_ORIG state is UNLOCKED. 10 <sub>B</sub> UCB OTP4_ORIG state is CONFIRMED. 11 <sub>B</sub> UCB OTP4_ORIG state is ERRORRED.

Field	Bits	Type	Description
PROINOTP50	11:10	rh	<b>UCB OTP5_ORIG Confirmation</b> This bit reflects the confirmed state of UCB OTP5_ORIG. $00_B$ UCB OTP5_ORIG state is UNREAD. $01_B$ UCB OTP5_ORIG state is UNLOCKED. $10_B$ UCB OTP5_ORIG state is CONFIRMED. $11_B$ UCB OTP5_ORIG state is ERRORRED.
PROINOTP60	13:12	rh	<b>UCB OTP6_ORIG Confirmation</b> This bit reflects the confirmed state of UCB OTP6_ORIG. $00_B$ UCB OTP6_ORIG state is UNREAD. $01_B$ UCB OTP6_ORIG state is UNLOCKED. $10_B$ UCB OTP6_ORIG state is CONFIRMED. $11_B$ UCB OTP6_ORIG state is ERRORRED.
PROINOTP70	15:14	rh	<b>UCB OTP7_ORIG Confirmation</b> This bit reflects the confirmed state of UCB OTP7_ORIG. $00_B$ UCB OTP7_ORIG state is UNREAD. $01_B$ UCB OTP7_ORIG state is UNLOCKED. $10_B$ UCB OTP7_ORIG state is CONFIRMED. $11_B$ UCB OTP7_ORIG state is ERRORRED.
PROINOTP0C	17:16	rh	<b>UCB OTP0_COPY Confirmation</b> This bit reflects the confirmed state of UCB OTP0_COPY. $00_B$ UCB OTP0_COPY state is UNREAD. $01_B$ UCB OTP0_COPY state is UNLOCKED. $10_B$ UCB OTP0_COPY state is CONFIRMED. $11_B$ UCB OTP0_COPY state is ERRORRED.
PROINOTP1C	19:18	rh	<b>UCB OTP1_COPY Confirmation</b> This bit reflects the confirmed state of UCB OTP1_COPY. $00_B$ UCB OTP1_COPY state is UNREAD. $01_B$ UCB OTP1_COPY state is UNLOCKED. $10_B$ UCB OTP1_COPY state is CONFIRMED. $11_B$ UCB OTP1_COPY state is ERRORRED.
PROINOTP2C	21:20	rh	<b>UCB OTP2_COPY Confirmation</b> This bit reflects the confirmed state of UCB OTP2_COPY. $00_B$ UCB OTP2_COPY state is UNREAD. $01_B$ UCB OTP2_COPY state is UNLOCKED. $10_B$ UCB OTP2_COPY state is CONFIRMED. $11_B$ UCB OTP2_COPY state is ERRORRED.
PROINOTP3C	23:22	rh	<b>UCB OTP3_COPY Confirmation</b> This bit reflects the confirmed state of UCB OTP3_COPY. $00_B$ UCB OTP3_COPY state is UNREAD. $01_B$ UCB OTP3_COPY state is UNLOCKED. $10_B$ UCB OTP3_COPY state is CONFIRMED. $11_B$ UCB OTP3_COPY state is ERRORRED.

Field	Bits	Type	Description
<b>PROINOTP4C</b>	25:24	rh	<b>UCB OTP4 COPY Confirmation</b> This bit reflects the confirmed state of UCB OTP4_COPY. $00_B$ UCB OTP4_COPY state is UNREAD. $01_B$ UCB OTP4_COPY state is UNLOCKED. $10_B$ UCB OTP4_COPY state is CONFIRMED. $11_B$ UCB OTP4_COPY state is ERRORED.
<b>PROINOTP5C</b>	27:26	rh	<b>UCB OTP5 COPY Confirmation</b> This bit reflects the confirmed state of UCB OTP5_COPY. $00_B$ UCB OTP5_COPY state is UNREAD. $01_B$ UCB OTP5_COPY state is UNLOCKED. $10_B$ UCB OTP5_COPY state is CONFIRMED. $11_B$ UCB OTP5_COPY state is ERRORED.
<b>PROINOTP6C</b>	29:28	rh	<b>UCB OTP6 COPY Confirmation</b> This bit reflects the confirmed state of UCB OTP6_COPY. $00_B$ UCB OTP6_COPY state is UNREAD. $01_B$ UCB OTP6_COPY state is UNLOCKED. $10_B$ UCB OTP6_COPY state is CONFIRMED. $11_B$ UCB OTP6_COPY state is ERRORED.
<b>PROINOTP7C</b>	31:30	rh	<b>UCB OTP7 COPY Confirmation</b> This bit reflects the confirmed state of UCB OTP7_COPY. $00_B$ UCB OTP7_COPY state is UNREAD. $01_B$ UCB OTP7_COPY state is UNLOCKED. $10_B$ UCB OTP7_COPY state is CONFIRMED. $11_B$ UCB OTP7_COPY state is ERRORED.

### 6.5.3.2.3 Flash Error Registers

#### Enable Error Interrupt Control Register

**HF\_EER**

**Enable Error Interrupt Control Register (0000030<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EOBM</b>									<b>RES</b>						
rw								r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
											<b>EVER M</b>	<b>PVER M</b>	<b>PROE RM</b>	<b>SQER M</b>	<b>OPER M</b>
											rw	rw	rw	rw	rw

Field	Bits	Type	Description
<b>OPERM</b>	0	rw	<b>Operation Error Interrupt Mask</b> $0_B$ Interrupt disabled. $1_B$ Flash interrupt because Operation Error is enabled.

Field	Bits	Type	Description
<b>SQERM</b>	1	rw	<b>Command Sequence Error Interrupt Mask</b> $0_B$ Interrupt disabled. $1_B$ Flash interrupt because Sequence Error is enabled.
<b>PROERM</b>	2	rw	<b>Protection Error Interrupt Mask</b> $0_B$ Interrupt disabled. $1_B$ Flash interrupt because Protection Error is enabled.
<b>PVERM</b>	3	rw	<b>Program Verify Error Interrupt Mask</b> $0_B$ Interrupt disabled. $1_B$ Flash interrupt because Program Verify Error is enabled.
<b>EVERM</b>	4	rw	<b>Erase Verify Error Interrupt Mask</b> $0_B$ Interrupt disabled. $1_B$ Flash interrupt because Erase Verify Error is enabled.
<b>RES</b>	30:5	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>EOBM</b>	31	rw	<b>End of Busy Interrupt Mask</b> $0_B$ Interrupt disabled. $1_B$ EOB interrupt is enabled.

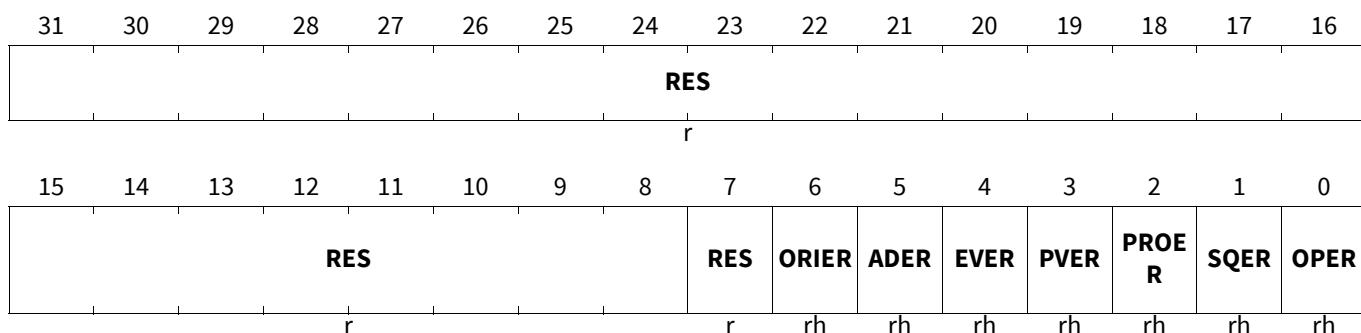
### Error Status Register

#### HF\_ERRSR

##### Error Status Register

(0000034<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>OPER</b>	0	rh	<b>Flash Operation Error</b>  Note: Cleared with system reset.  $0_B$ No operation error. $1_B$ Flash array operation aborted, because of a Flash array failure, e.g. an ECC error in FSI SRAM.

Field	Bits	Type	Description
SQER	1	rh	<p><b>Command Sequence Error</b>            A sequence error is not indicated if the Reset to Read command aborts a command sequence.</p> <p><b>Note:</b> Cleared with application reset, commands “Reset to Read” and “Clear Status” or writing <a href="#">HF_CLRE.CSQER = 1<sub>B</sub></a>.</p> <p>0<sub>B</sub> No sequence error.            1<sub>B</sub> Command state machine operation unsuccessful because of improper address or command sequence.</p>
PROER	2	rh	<p><b>Protection Error</b>            A Protection Error is reported e.g. because of a not allowed command, for example an Erase or Write Page command addressing a locked sector, or because of wrong password(s) in a protected command sequence such as “Disable Read Protection”.</p> <p><b>A Protection Error is also reported if the safety protection prevented a program/erase operation in Flash.</b></p> <p><b>Note:</b> Cleared with application reset, with the command “Clear Status” or writing <a href="#">HF_CLRE.CPROER = 1<sub>B</sub></a>.</p> <p>0<sub>B</sub> No protection error.            1<sub>B</sub> Protection error.</p>
PVER	3	rh	<p><b>Program Verify Error</b>            A verify error was reported on completion of a Flash program operation</p> <p><b>Note:</b> Cleared with application reset, with the command “Clear Status” or writing <a href="#">HF_CLRE.CPVER = 1<sub>B</sub></a>.</p> <p>0<sub>B</sub> The page is correctly programmed. All bits have full expected quality.            1<sub>B</sub> A program verify error has been detected. Full quality of all bits cannot be guaranteed.</p>
EVER	4	rh	<p><b>Erase Verify Error</b>            A verify error was reported on completion of a Flash erase operation.</p> <p><b>Note:</b> Cleared with application reset, with the command “Clear Status” or writing <a href="#">HF_CLRE.CEVER = 1<sub>B</sub></a>.</p> <p>0<sub>B</sub> The sector is correctly erased. All erased bits have full expected quality.            1<sub>B</sub> An erase verify error has been detected. Full quality erased bits cannot be guaranteed.</p>

Field	Bits	Type	Description
<b>ADER</b>	5	rh	<p><b>SRI Bus Address ECC Error</b>            This flag is set when the DMU detects an ECC error in the address phase bus transaction on the SRI bus.</p> <p><i>Note:</i> Cleared with application reset or writing <b>HF_CLRE.CADER</b> = <math>1_B</math>.</p> <p><math>0_B</math> No SRI address error detected.  <math>1_B</math> SRI address error detected.</p>
<b>ORIER</b>	6	rh	<p><b>Original Error</b>            If a UCB contains both ORIG and COPY confirmation codes and during startup an ERRORED value or an uncorrectable ECC error is detected in the ORIG confirmation code then the original error flag is set by hardware.</p> <p><i>Note:</i> Cleared with application reset.</p> <p><math>0_B</math> No original error detected.  <math>1_B</math> Original error detected.</p>
<b>RES</b>	7, 31:8	r	<b>Reserved</b>

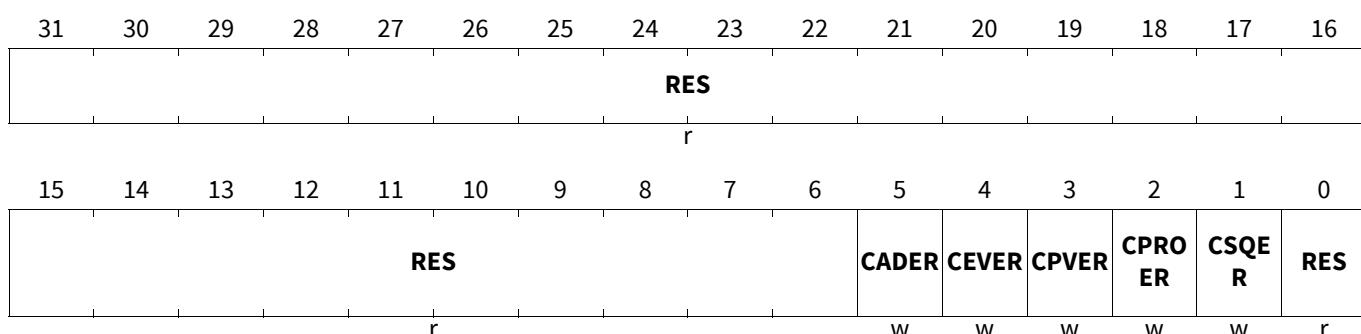
### Clear Error Register

#### HF\_CLRE

##### Clear Error Register

(0000038<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>RES</b>	0, 31:6	r	<p><b>Reserved</b>            Always read as 0; should be written with 0.</p>
<b>CSQER</b>	1	w	<p><b>Clear Command Sequence Error</b>  <math>0_B</math> No action  <math>1_B</math> Clear Command Sequence Error Flag DMU_HF_ERRSR.SQER</p>
<b>CPOER</b>	2	w	<p><b>Clear Protection Error</b>  <math>0_B</math> No action  <math>1_B</math> Clear Protection Error Flag DMU_HF_ERRSR.PROER</p>

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>CPVER</b>	3	w	<b>Clear Program Verify Error</b> 0 <sub>B</sub> No action 1 <sub>B</sub> Clear Program Verify Error Flag DMU_HF_ERRSR.PVER
<b>CEVER</b>	4	w	<b>Clear Erase Verify Error</b> 0 <sub>B</sub> No action 1 <sub>B</sub> Clear Erase Verify Error Flag DMU_HF_ERRSR.EVER
<b>CADER</b>	5	w	<b>Clear SRI Bus Address ECC Error</b> 0 <sub>B</sub> No action 1 <sub>B</sub> Clear SRI Bus Address ECC Error Flag DMU_HF_ERRSR.ADER

#### **6.5.3.2.4 Data Flash Bank 0 ECC Registers**

## DF0 ECC Read Register

The ECC Read Register must store the ECC checksum read during the last DF0 (including CFS and UCB) and DF1 (when DF1 is configured as not HSM\_exclusive) NVM read access when the read is initiated via the DMU SRI slave interface.

HF ECCR

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>RCODE</b>	21:0	rh	<b>Error Correction Read Code</b> ECC checksum read during the last NVM read access.
<b>RES</b>	23:22, 31:24	r	<b>Reserved</b>

## DF0 ECC Status Register

The ECC Status Register must capture ECC errors detected during the last DF0 (including CFS and UCB) and DF1 (when DF1 is configured as not HSM\_exclusive) NVM read access when the read is initiated via the DMU SRI slave interface.

### HF\_ECCS

#### DF0 ECC Status Register

(0000044<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
						<b>ABLAN KA</b>	<b>RES</b>	<b>AERAN Y</b>			<b>RES</b>		<b>AERM</b>	<b>AER3</b>	<b>AER2</b>	<b>AER1</b>
r					rh	rX	rh		r		rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
						<b>BLANK A</b>	<b>RES</b>	<b>ERRA NY</b>			<b>RES</b>		<b>ERRM</b>	<b>ERR3</b>	<b>ERR2</b>	<b>ERR1</b>
r					rh	rX	rh		r		rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
<b>ERR1</b>	0	rh	<p><b>Read Access Single Bit ECC Error</b>            The flag reports a single bit ECC failure during the last NVM read access.</p> <p><i>Note:</i> Reset by hardware when <b>HF_ECCC.CLR</b> is written to 11<sub>B</sub>.</p> <p>0<sub>B</sub> No single bit ECC failure occurred.            1<sub>B</sub> A single bit ECC failure occurred.</p>
<b>ERR2</b>	1	rh	<p><b>Read Access Double Bit ECC Error</b>            The flag reports a double bit ECC failure during the last NVM read access.</p> <p><i>Note:</i> Reset by hardware when <b>HF_ECCC.CLR</b> is written to 11<sub>B</sub>.</p> <p>0<sub>B</sub> No double bit ECC failure occurred.            1<sub>B</sub> A double bit ECC failure occurred.</p>
<b>ERR3</b>	2	rh	<p><b>Read Access Triple Bit ECC Error</b>            The flag reports a triple bit ECC failure during the last NVM read access.</p> <p><i>Note:</i> Reset by hardware when <b>HF_ECCC.CLR</b> is written to 11<sub>B</sub>.</p> <p>0<sub>B</sub> No triple bit ECC failure occurred.            1<sub>B</sub> A triple bit ECC failure occurred.</p>
<b>ERRM</b>	3	rh	<p><b>Read Access Multi-bit ECC Error</b>            The flag reports multi bit ECC failure during the last NVM read access.</p> <p><i>Note:</i> Reset by hardware when <b>HF_ECCC.CLR</b> is written to 11<sub>B</sub>.</p> <p>0<sub>B</sub> No multi bit ECC failure occurred.            1<sub>B</sub> Multi bit ECC failure occurred.</p>
<b>RES</b>	6:4, 15:10, 22:20, 31:26	r	<p><b>Reserved</b>            Always read as 0; should be written with 0.</p>

Field	Bits	Type	Description
ERRANY	7	rh	<p><b>Any Read Access ECC Error</b>            The flag reports any ECC failure during the last NVM read access.</p> <p><i>Note:</i> Reset by hardware when <a href="#">HF_ECCC.CLR</a> is written to <math>11_B</math>.</p> <p><math>0_B</math> No ECC failure occurred.  <math>1_B</math> ECC failure occurred.</p>
RES	8, 24	rX	<b>Reserved</b>
BLANKA	9	rh	<p><b>Read Access Blank Analog</b>            The flag reports that all read data cells have sufficient high current: a program of new data without prior erase is possible. Under certain operation history, a valid complement data entry may also appear as blank. Data qualifiers like headers or footers, which are usually used in EEPROM emulation, can be used to distinguish expected valid data from unknown data fields. Only blank failures in Complement Sensing mode are reported in this flag and is intended for use only in this mode.</p> <p><i>Note:</i> Reset by hardware when <a href="#">HF_ECCC.CLR</a> is written to <math>11_B</math>.</p> <p><math>0_B</math> Read data cells are not erased.  <math>1_B</math> Read data cells have sufficient high current: a program of new data without prior erase is possible.</p>
AER1	16	rh	<p><b>Accumulated Single Bit ECC Errors</b>            The status flag accumulates single bit failures during NVM read operations.</p> <p><i>Note:</i> Reset by hardware when <a href="#">HF_ECCC.CLR</a> is written to <math>11_B</math>.</p> <p><math>0_B</math> No single bit ECC failure occurred.  <math>1_B</math> At least one single bit ECC failure occurred.</p>
AER2	17	rh	<p><b>Accumulated Double Bit ECC Errors</b>            The status flag accumulates double bit failures during NVM read operations.</p> <p><i>Note:</i> Reset by hardware when <a href="#">HF_ECCC.CLR</a> is written to <math>11_B</math>.</p> <p><math>0_B</math> No double bit ECC failure occurred.  <math>1_B</math> At least one double bit ECC failure occurred.</p>
AER3	18	rh	<p><b>Accumulated Triple Bit ECC Errors</b>            The status flag accumulates triple bit failures during NVM read operations.</p> <p><i>Note:</i> Reset by hardware when <a href="#">HF_ECCC.CLR</a> is written to <math>11_B</math>.</p> <p><math>0_B</math> No triple bit ECC failure occurred.  <math>1_B</math> At least one triple bit ECC failure occurred.</p>

Field	Bits	Type	Description
<b>AERM</b>	19	rh	<p><b>Accumulated Multi-bit ECC Errors</b>            The status bit accumulates multi bit failures during NVM read accesses.</p> <p><i>Note:</i> Reset by hardware when <b>HF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> No multi bit ECC failure occurred.  <math>1_B</math> Multi bit ECC failure occurred.</p>
<b>AERANY</b>	23	rh	<p><b>Accumulated Any Read Access ECC Error</b>            The status bit accumulates ECC failures during NVM read accesses.</p> <p><i>Note:</i> Reset by hardware when <b>HF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> No ECC failure occurred.  <math>1_B</math> ECC failure occurred.</p>
<b>ABLANKA</b>	25	rh	<p><b>Accumulated Blank Analog</b>            The flag accumulates analog evaluated blank failures during NVM read accesses. It reports that all read data cells have sufficient high current: a program of new data without prior erase is possible. Under certain operation history, a valid complement data entry may also appear as blank. Data qualifiers like headers or footers, which are usually used in EEPROM emulation, can be used to distinguish expected valid data from unknown data fields. Only blank failures in Complement Sensing mode are reported in this flag and is intended for use only in this mode.</p> <p><i>Note:</i> Reset by hardware when <b>HF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> Read data cells are not erased.  <math>1_B</math> Read data cells have sufficient high current: a program of new data without prior erase is possible.</p>

## DF0 ECC Control Register

### HF\_ECCC

#### DF0 ECC Control Register

(0000048<sub>H</sub>)

Application Reset Value: C000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>TRAPDIS</b>	<b>ECCCORDIS</b>									<b>RES</b>					
rw	rw								r						w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								<b>RES</b>						<b>CLR</b>	
								r							w

Field	Bits	Type	Description
CLR	1:0	w	<b>Clear ECC status bits</b> $00_B$ No action. ... $10_B$ No action. $11_B$ Clear DMU_HF_ECCS status bits.
RES	27:2	r	<b>Reserved</b> Always read as 0; should be written with 0.
ECCCORDIS	29:28	rw	<b>Host Command Interface ECC Correction Disable</b> $00_B$ <b>Enabled</b> , ECC correction is enabled for the DF0 read path and if DF1 is configured as not HSM_exclusive, then for the DF1 read path. $01_B$ <b>Enabled</b> , ECC correction is enabled for the DF0 read path and if DF1 is configured as not HSM_exclusive, then for the DF1 read path. $10_B$ <b>Enabled</b> , ECC correction is enabled for the DF0 read path and if DF1 is configured as not HSM_exclusive, then for the DF1 read path. $11_B$ <b>Disabled</b> , ECC correction is disabled for the DF0 read path and if DF1 is configured as not HSM_exclusive, then for the DF1 read path.
TRAPDIS	31:30	rw	<b>Host Command Interface Uncorrectable ECC Bit Error Trap Disable</b> $00_B$ For DF0 and DF1 (when DF1 is configured as not HSM_exclusive), if an uncorrectable ECC error occurs then a bus error trap is generated. ... $10_B$ For DF0 and DF1 (when DF1 is configured as not HSM_exclusive), if an uncorrectable ECC error occurs then a bus error trap is generated. $11_B$ For DF0 and DF1 (when DF1 is configured as not HSM_exclusive), the uncorrectable ECC error trap is disabled.

### DF0 ECC Write Register

The ECC Write Register contains bits for disabling the ECC encoding for PFLASH Banks and DF0.

When disabling the ECC encoding for a PFLASH Bank with **HF\_ECCW.PECENCDIS** = ' $11_B$ ' the ECC code for the next 256-bit data block transferred from DMU to the Flash assembly buffer is taken from **HF\_ECCW.WCODE**.

When disabling the ECC encoding for DF0 with **HF\_ECCW.DECENCDIS** = ' $11_B$ ' the ECC code for the next 64-bit data block transferred from DMU to the Flash assembly buffer is taken from **HF\_ECCW.WCODE**.

If **HF\_ECCW.PECENCDIS** or **HF\_ECCW.DECENCDIS** is set to ' $11_B$ ' then the "Write Burst" command sequence will cause unpredictable results and must not be used.

**Attention:** After reading data with disabled ECC correction data buffers it is recommended to perform a reset to resume normal operation with ECC correction

**Attention:** Software should only set one of **HF\_ECCW.PECENCDIS** or **HF\_ECCW.DECENCDIS** to ' $11B$ '.

**HF\_ECCW****DF0 ECC Write Register**(000004C<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>DECENCDIS</b>	<b>PECENCDIS</b>						<b>RES</b>								<b>WCODE</b>
rw	rw					r									rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								<b>WCODE</b>							rw

Field	Bits	Type	Description
<b>WCODE</b>	21:0	rw	<b>Error Correction Write Code</b> 22-bit ECC code for the current 64-bit (for DFLASH) or 256-bit (for PFLASH) write buffer to be written into the assembly buffer instead of a generated ECC.
<b>RES</b>	27:22	r	<b>Reserved</b> Always read as 0.
<b>PECENCDIS</b>	29:28	rw	<b>PFLASH ECC Encoding Disable</b> 00 <sub>B</sub> The ECC code is automatically calculated. ... 10 <sub>B</sub> The ECC code is automatically calculated. 11 <sub>B</sub> The ECC code is taken from WCODE.
<b>DECENCDIS</b>	31:30	rw	<b>DFLASH ECC Encoding Disable</b> 00 <sub>B</sub> The ECC code is automatically calculated. ... 10 <sub>B</sub> The ECC code is automatically calculated. 11 <sub>B</sub> The ECC code is taken from WCODE.

**6.5.3.2.5 Data Flash Bank 0 Mode Control Registers****DF0 User Mode Control**

The DF0 Protection Configuration User Mode Control Register is loaded from UCB during startup.

**HF\_PROCONUSR****DF0 User Mode Control**(0000074<sub>H</sub>)Reset Value: [Table 155](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
															<b>UCB</b>
															rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								<b>UCB</b>							<b>MODE</b>
															rh

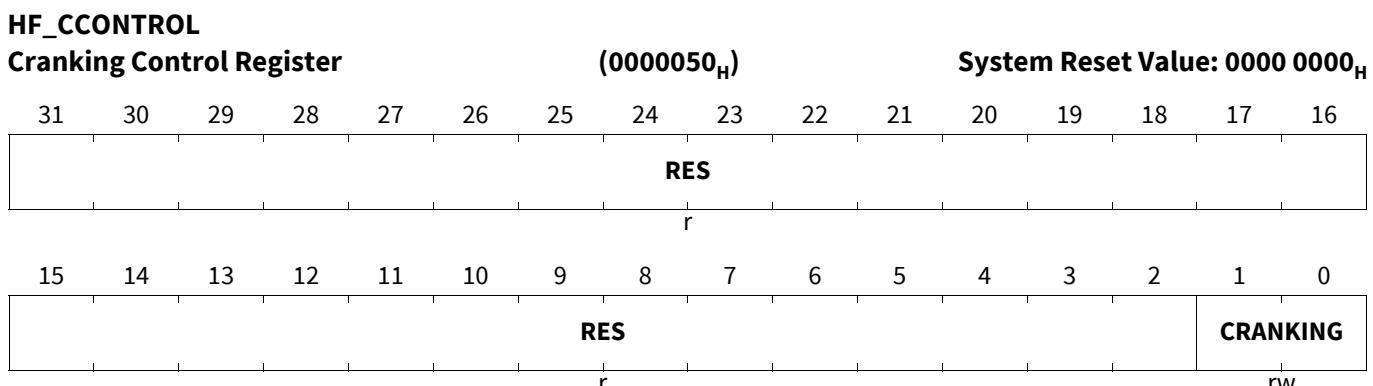
Field	Bits	Type	Description
MODE	1:0	rh	<b>DF0 User Mode Control</b> Configures the DF0 mode when the user has control. $00_B$ Single Ended. $01_B$ Complement Sensing. ... $11_B$ Complement Sensing.
UCB	31:2	rh	<b>Reserved for UCB</b> Deliver the corresponding content of UCB.

Table 155 Reset Values of HF\_PROCONUSR

Reset Type	Reset Value	Note
Application Reset	$0000\ 0000_H$	
CFS Value	$0000\ 0000_H$	

### 6.5.3.2.6 Power Mode Registers

#### Cranking Control Register



Field	Bits	Type	Description
CRANKING	1:0	rw	<b>Cranking Mode Control</b> This bit field determines Cranking mode. $00_B$ Cranking mode not selected ... $10_B$ Cranking mode not selected $11_B$ Cranking mode selected
RES	31:2	r	<b>Reserved</b> Always read as 0; should be written with 0.

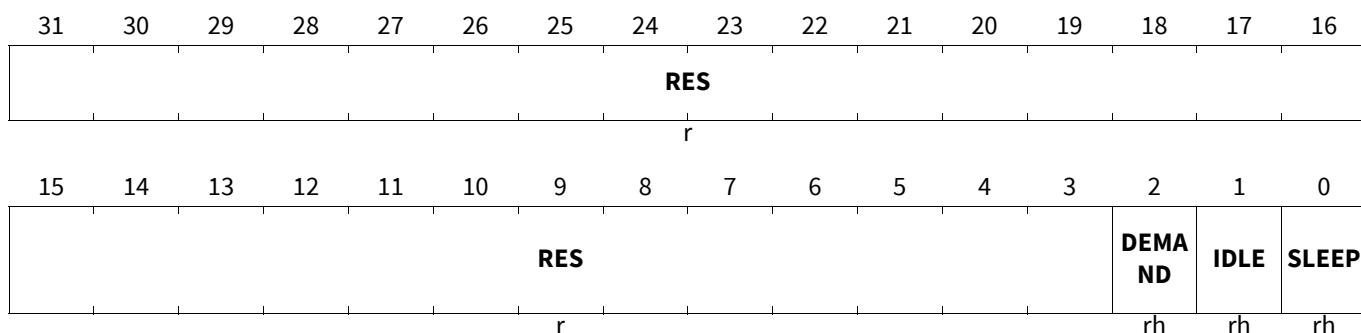
## Power Status Register

### HF\_PSTATUS

#### Power Status Register

(0000060<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>SLEEP</b>	0	rh	<b>Sleep Mode</b>  $0_B$ DMU is not in Sleep Mode $1_B$ DMU is in Sleep Mode
<b>IDLE</b>	1	rh	<b>Dynamic Idle</b>  $0_B$ PFLASH is not in Dynamic Idle mode. $1_B$ PFLASH is in Dynamic Idle mode.
<b>DEMAND</b>	2	rh	<b>Demand</b>  <i>Note:</i> For Cranking Mode, HF_PSTATUS.DEMAND = 1 <sub>B</sub>  $0_B$ PFLASH prefetch buffers enabled. $1_B$ PFLASH prefetch buffers disabled.
<b>RES</b>	31:3	r	<b>Reserved</b> Always read as 0; should be written with 0.

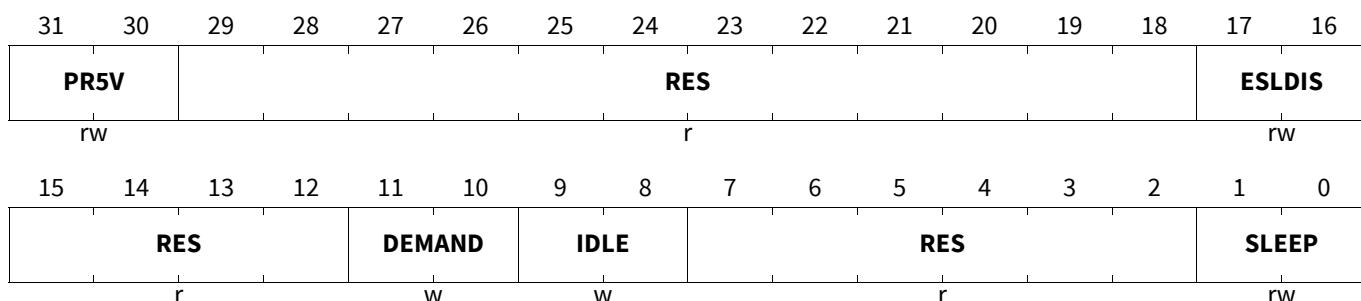
## Power Control Register

### HF\_PCONTROL

#### Power Control Register

(0000064<sub>H</sub>)

Application Reset Value: 0003 0000<sub>H</sub>



Field	Bits	Type	Description
<b>SLEEP</b>	1:0	rw	<b>Sleep Mode Control</b> This bit field determines the Sleep mode. 00 <sub>B</sub> Normal operation or Wake up ... 10 <sub>B</sub> Normal operation or Wake up 11 <sub>B</sub> Sleep mode is requested
<b>RES</b>	7:2, 15:12, 29:18	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>IDLE</b>	9:8	w	<b>Dynamic Idle Enable</b> 00 <sub>B</sub> Disable Dynamic Idle (DMU_HF_PSTATUS.IDLE = 0 <sub>B</sub> ). ... 10 <sub>B</sub> Disable Dynamic Idle (DMU_HF_PSTATUS.IDLE = 0 <sub>B</sub> ). 11 <sub>B</sub> Enable Dynamic Idle (DMU_HF_PSTATUS.IDLE = 1 <sub>B</sub> ).
<b>DEMAND</b>	11:10	w	<b>Demand Enable</b> 00 <sub>B</sub> Disable Demand Mode (DMU_HF_PSTATUS.DEMAND = 0 <sub>B</sub> ). ... 10 <sub>B</sub> Disable Demand Mode (DMU_HF_PSTATUS.DEMAND = 0 <sub>B</sub> ). 11 <sub>B</sub> Enable Demand Mode (DMU_HF_PSTATUS.DEMAND = 1 <sub>B</sub> ).
<b>ESLDIS</b>	17:16	rw	<b>External Sleep Mode Request Disable</b> Used for Sleep Mode control. 00 <sub>B</sub> Sleep Mode request is enabled. 01 <sub>B</sub> Sleep mode request is disabled. Sleep Mode cannot be entered. 10 <sub>B</sub> Sleep Mode request is disabled. Sleep Mode cannot be entered. 11 <sub>B</sub> Sleep Mode request is disabled. Sleep Mode cannot be entered.
<b>PR5V</b>	31:30	rw	<b>Programming Supply 5V</b> Selects the supply for programming. 00 <sub>B</sub> <b>P3V</b> , The programming voltage is internally generated. ... 10 <sub>B</sub> <b>P3V</b> , The programming voltage is internally generated. 11 <sub>B</sub> <b>P5V</b> , As programming voltage the external 5 V supply is used.

**PFLASH Wait Cycle Register****HF\_PWAIT****PFLASH Wait Cycle Register**(00000068<sub>H</sub>)System Reset Value: 0716 040B<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES				CECC				RES		CFLASH					
				r				rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				RECC				RES		RFLASH					
				r				rw							

Field	Bits	Type	Description
RFLASH	5:0	rw	<b>Operation Mode</b> <b>PFLASH Wait Cycles</b> This bit field defines the number of SRI clock cycles for a PFLASH Bank read access. $00_H$ Read access takes 2 fSRI clock cycles. $01_H$ Read access takes 2 fSRI clock cycles ... $3F_H$ Read access takes 64 fSRI clock cycles
RES	7:6, 15:11, 23:22, 31:27	r	<b>Reserved</b> Always read as 0; should be written with 0.
RECC	10:8	rw	<b>Operation Mode</b> <b>PFLASH Error Correction Cycles</b> This bit field defines the number of SRI clock cycles for the PFLASH Bank ECC correction. $000_B$ Error correction takes 1 fSRI clock cycle. $001_B$ Error correction takes 2 fSRI clock cycles ... $111_B$ Error correction takes 8 fSRI clock cycles
CFLASH	21:16	rw	<b>Cranking Mode</b> <b>PFLASH Wait Cycles</b> This bit field defines the number of SRI clock cycles for a PFLASH Bank read access. $00_H$ Read access takes 2 fSRI clock cycles. $01_H$ Read access takes 2 fSRI clock cycles ... $3F_H$ Read access takes 64 fSRI clock cycles
CECC	26:24	rw	<b>Cranking Mode</b> <b>PFLASH Error Correction Cycles</b> This bit field defines the number of SRI clock cycles for the PFLASH Bank ECC correction. $000_B$ Error correction takes 1 fSRI clock cycle. $001_B$ Error correction takes 2 fSRI clock cycles ... $111_B$ Error correction takes 8 fSRI clock cycles

### DFLASH Wait Cycle Register

#### HF\_DWAIT

**DFLASH Wait Cycle Register** **(000006C<sub>H</sub>)** **System Reset Value: 0004 000B<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES												RECC			
r															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								RFLASH							
r															rw

Field	Bits	Type	Description
RFLASH	7:0	rw	<b>Operation Mode</b> <b>DFLASH Wait Cycles for Flash</b> This bit field defines the number of FSI clock cycles for a DFLASH Bank read access. 00 <sub>H</sub> Read access takes 2 fFSI clock cycles. 01 <sub>H</sub> Read access takes 2 fFSI clock cycles ... FF <sub>H</sub> Read access takes 256 fFSI clock cycles
RES	15:8, 31:19	r	<b>Reserved</b> Always read as 0; should be written with 0.
RECC	18:16	rw	<b>Operation Mode</b> <b>DFLASH Error Correction Cycles</b> This bit field defines the number of FSI clock cycles for the DFLASH Bank ECC correction. 000 <sub>B</sub> Error correction takes 1 fFSI clock cycle. 001 <sub>B</sub> Error correction takes 2 fFSI clock cycles ... 111 <sub>B</sub> Error correction takes 8 fFSI clock cycles

### 6.5.3.2.7 PFLASH Protection Configuration

#### PFLASH Protection Configuration

**HF\_PROCONPF**  
**PFLASH Protection Configuration** **(0000080<sub>H</sub>)** **Reset Value: Table 156**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>RPRO</b>		UCB													
rh															rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB															
rh															

Field	Bits	Type	Description
<b>UCB</b>	30:0	rh	<b>Reserved for UCB</b> Deliver the corresponding content of UCB.
<b>RPRO</b>	31	rh	<b>Read Protection Configuration</b> This bit indicates whether read protection is configured for PFLASH sectors. 0 <sub>B</sub> No read protection configured 1 <sub>B</sub> Read protection and write protection is configured.

**Table 156 Reset Values of HF\_PROCONPF**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### 6.5.3.2.8 Tuning Protection Configuration

#### Tuning Protection Configuration

**HF\_PROCONT**  
**Tuning Protection Configuration** **(00000084<sub>H</sub>)** **Reset Value: Table 157**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB															
								rh							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB															
								rh			rh				rh

Field	Bits	Type	Description
<b>TP</b>	0	rh	<b>Tuning Protection</b> This bit indicates whether tuning protection is installed or not. 0 <sub>B</sub> Tuning protection is not configured. 1 <sub>B</sub> Tuning protection is configured and installed, if correctly confirmed.
<b>UCB</b>	7:1, 15:10, 31:24	rh	<b>Reserved for UCB</b> Deliver the corresponding content of UCB.

Field	Bits	Type	Description
<b>BML</b>	9:8	rh	<b>Boot Mode Lock</b> Used by the SSW to restrict the boot mode selection. 00 <sub>B</sub> Boot flow with standard evaluation of boot headers. 01 <sub>B</sub> Restricted boot flow, never evaluating HWCFG pins and without fallback to boot loader. ... 11 <sub>B</sub> Restricted boot flow, never evaluating HWCFG pins and without fallback to boot loader.
<b>SWAPEN</b>	17:16	rh	<b>Enable SOTA mode</b> This field enables the entry into "Software update Over the Air(SOTA) mode". In this mode, an alternate PFLASH address map can be selected. Please refer to the SOTA section of the Introduction chapter for more details. 00 <sub>B</sub> <b>Disabled</b> , SOTA mode disabled. ... 10 <sub>B</sub> <b>Disabled</b> , SOTA mode disabled. 11 <sub>B</sub> <b>Enabled</b> , SOTA mode enabled.
<b>CPUxDDIS (x=0-5)</b>	x+18	rh	<b>Disable direct LPB access</b> Disable direct LPB access by the CPU to the Local PFlash Bank (LPB). 0 <sub>B</sub> Direct LPB access is enabled. 1 <sub>B</sub> Direct LPB access is disabled.

**Table 157 Reset Values of HF\_PROCONTP**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### 6.5.3.2.9 DFLASH Protection Configuration

The protection configuration is indicated with registers PROCONDF, PROCONRAM and PROCONDBG.

#### DFLASH Protection Configuration

**HF\_PROCONDF**  
**DFLASH Protection Configuration** **(00000088<sub>H</sub>)** **Reset Value: Table 158**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>RPRO</b>	<b>UCB</b>		<b>ESR0CNT</b>												
rh	rh								rh						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>CAP3E N</b>	<b>CAP2E N</b>	<b>CAP1E N</b>	<b>CAP0E N</b>	<b>APRE N</b>	<b>MODE</b>	<b>OSCCFG</b>	<b>AMPCTL</b>	<b>HYSCTL</b>	<b>HYSEN</b>	<b>UCB</b>	<b>L</b>				
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	

Field	Bits	Type	Description
L	0	rh	<p><b>DF0_EEPROM Locked for Write Protection</b>            This bit indicates whether the DFLASH sectors EEPROMx are write protected.  <math>0_B</math> No write protection is configured.  <math>1_B</math> Write protection is configured.</p>
UCB	2:1, 30:28	rh	<p><b>Reserved</b>            Deliver the corresponding content of UCB.</p>
HYSEN	3	rh	<p><b>Hysteresis enable</b>            When enabled by OSCCFG these fields are copied to SCU_OSCCON.HYSEN  <math>0_B</math> Hysteresis is disabled  <math>1_B</math> Hysteresis is enabled</p>
HYSCTL	5:4	rh	<p><b>Hysteresis Control</b>            When enabled by OSCCFG these fields are copied to SCU_OSCCON.HYSCTL  <math>00_B</math> Hysteresis setting 1  <math>01_B</math> Hysteresis setting 2  <math>10_B</math> Hysteresis setting 3  <math>11_B</math> Hysteresis setting 4</p>
AMPCTL	7:6	rh	<p><b>Amplitude Control</b>            When enabled by OSCCFG these fields are copied to SCU_OSCCON.AMPCTL  <math>00_B</math> Amplitude control setting 1  <math>01_B</math> Amplitude control setting 2  <math>10_B</math> Amplitude control setting 3  <math>11_B</math> Amplitude control setting 4</p>
OSCCFG	8	rh	<p><b>OSC Configuration by SSW</b>            This bit indicates whether the oscillator configuration (fields: CAPxEN, APREN, MODE, HYSEN, HYSCTL, AMPCTL) are installed by the SSW in SCU_OSCCON.  <math>0_B</math> SSW does not install oscillator configuration values from this register into SCU_OSCCON.  <math>1_B</math> SSW configures oscillator with the configuration values in this register by installing the relevant contents into SCU_OSCCON.</p>
MODE	10:9	rh	<p><b>OSC Mode</b>            When enabled by OSCCFG this field is copied to SCU_OSCCON.MODE.</p>
APREN	11	rh	<p><b>OSC Amplitude Regulation Enable</b>            When enabled by OSCCFG this field is copied to SCU_OSCCON.APREN.</p>
CAPxEN (x=0-3)	x+12	rh	<p><b>OSC Capacitance x Enable (x=0-3)</b>            When enabled by OSCCFG these fields are copied to SCU_OSCCON.CAPxEN.</p>
ESR0CNT	27:16	rh	<p><b>ESR0 Prolongation Counter</b>            Used to configure the ESR0 delay. Evaluation by SSW.</p>

Field	Bits	Type	Description
RPRO	31	rh	<b>Read Protection Configuration</b> This bit indicates whether read protection is configured for DFLASH sectors. 0 <sub>B</sub> No read protection configured 1 <sub>B</sub> Read protection and write protection is configured.

**Table 158 Reset Values of HF\_PROCONDF**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

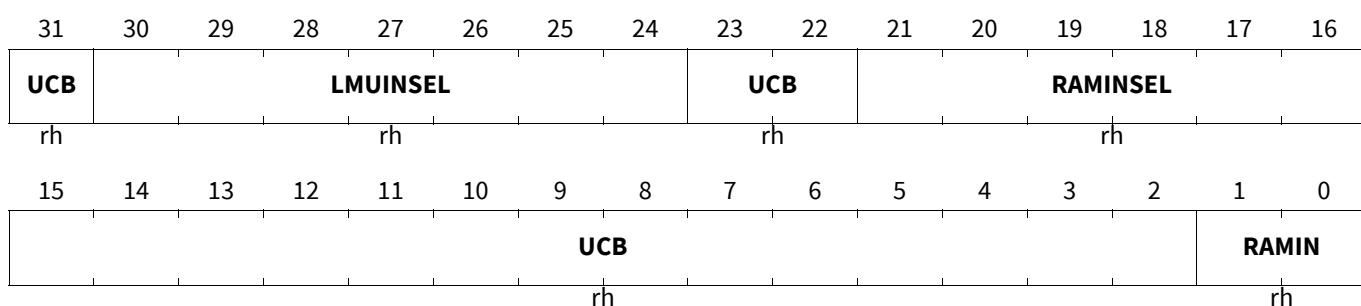
### RAM Configuration

#### HF\_PROCONRAM

##### RAM Configuration

(0000008C<sub>H</sub>)

**Reset Value: Table 159**



Field	Bits	Type	Description
RAMIN	1:0	rh	<b>RAM Initialization by SSW Control</b> These bits defined whether the RAMs selected by the field RAMINSEL are initialized. This field determines also if a RAM is initialized before MBIST access is granted. In all “Init_**” cases a RAM is initialized before MBIST access is enabled, in the “No_Init” case a RAM is not erased. This is independent of the memory selection with RAMINSEL for initialization during startup (see MTU chapter). 00 <sub>B</sub> <b>Init_All</b> , RAM initialization is performed after cold power-on- reset and warm power-on-reset. 01 <sub>B</sub> <b>Init_Warm</b> , RAM initialization is performed after warm power-on resets but not after cold power-on-reset (not recommended). 10 <sub>B</sub> <b>Init_Cold</b> , RAM initialization is performed after cold power-on- reset. 11 <sub>B</sub> <b>No_Init</b> , No RAM initialization is performed.
UCB	15:2, 23:22, 31	rh	<b>Reserved</b> Deliver the corresponding content of UCB.

Field	Bits	Type	Description
<b>RAMINSEL</b>	21:16	rh	<b>RAM Initialization Selection</b> These bits select which memories are initialized when the RAM initialization is configured with RAMIN. See <a href="#">Table 160</a> .
<b>LMUINSEL</b>	30:24	rh	<b>LMU Initialization Selection</b> These bits select which memories are initialized when the RAM initialization is configured with RAMIN. See <a href="#">Table 161</a> .

**Table 159 Reset Values of HF\_PROCONRAM**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**CPU RAM Initialisation Selection****Table 160 CPU RAM Initialization Selection<sup>1)</sup>**

RAMINSEL	Description
xxxxx0 <sub>B</sub>	RAMs (PSPR, DSPR, PCACHE and DCACHE) in CPU0 are selected for initialization.
xxxx0x <sub>B</sub>	RAMs (PSPR, DSPR, PCACHE and DCACHE) in CPU1 are selected for initialization.
xxx0xx <sub>B</sub>	RAMs (PSPR, DSPR, PCACHE and DCACHE) in CPU2 are selected for initialization.
xx0xxx <sub>B</sub>	RAMs (PSPR, DSPR, PCACHE and DCACHE) in CPU3 are selected for initialization.
x0xxxx <sub>B</sub>	RAMs (PSPR, DSPR, PCACHE and DCACHE) in CPU4 are selected for initialization.
0xxxxx <sub>B</sub>	RAMs (PSPR, DSPR, PCACHE and DCACHE) in CPU5 are selected for initialization.

1) Each bit with value '0' selects the corresponding memory for initialization.

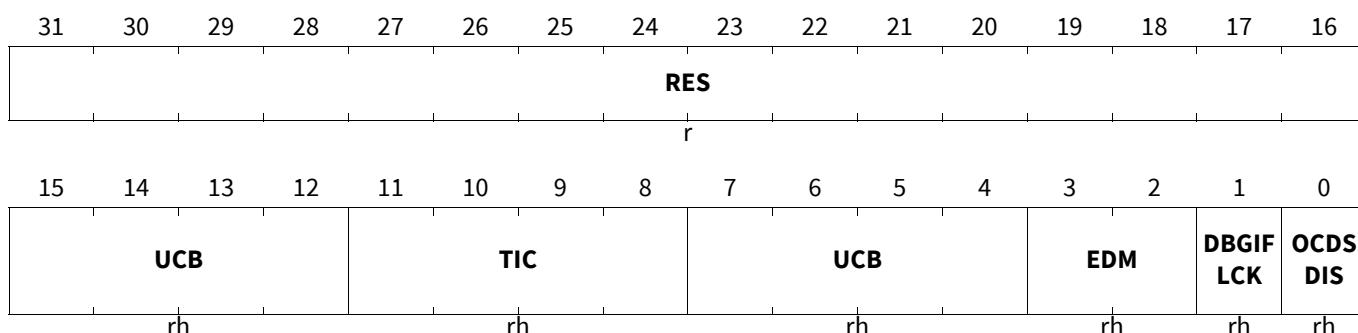
**LMU RAM Initialisation Selection****Table 161 LMU RAM Initialization Selection<sup>1)</sup>**

LMUINSEL	Description
xxxxxx0 <sub>B</sub>	CPU0 LMU is selected for initialization.
xxxxx0x <sub>B</sub>	CPU1 LMU is selected for initialization.
xxxx0xx <sub>B</sub>	CPU2 LMU is selected for initialization.
xxx0xxx <sub>B</sub>	CPU3 LMU is selected for initialization.
xx0xxxx <sub>B</sub>	CPU4 LMU is selected for initialization.
x0xxxxx <sub>B</sub>	CPU5 LMU is selected for initialization.
0xxxxxx <sub>B</sub>	Standalone LMU and DAMRAM is selected for initialization.

1) Each bit with value '0' selects the corresponding memory for initialization.

**Debug Interface Protection Configuration**

The debug interface and OCDS enable have a dedicated password protection logic.

**HF\_PROCONDBG****Debug Interface Protection Configuration (0000090<sub>H</sub>)****Reset Value: Table 162**

Field	Bits	Type	Description
<b>OCDSDIS</b>	0	rh	<b>OCDS Disabled</b> This bit indicates whether the OCDS is configured as locked. $0_B$ No OCDS lock configured in UCB_DBG. $1_B$ OCDS lock configured in UCB_DBG.
<b>DBGIFLCK</b>	1	rh	<b>Debug Interface Locked</b> This bit indicates whether the debug interface is configured as locked. $0_B$ No debug interface lock configured in UCB_DBG. $1_B$ Debug interface lock configured in UCB_DBG.
<b>EDM</b>	3:2	rh	<b>Entered Debug Mode</b> This bit indicates whether the debug interface has been opened via Destructive Debug Entry. Consequently the CAN and FlexRay operation is made impossible! $00_B$ “Debug not entered”, device operation not affected. ... $10_B$ “Debug not entered”, device operation not affected. $11_B$ “Debug entered”, CAN and FlexRay operation affected.
<b>UCB</b>	7:4, 15:12	rh	<b>Reserved</b> Deliver the corresponding content of UCB_DBG.
<b>TIC</b>	11:8	rh	<b>Tool Interface Control</b> DAP over CAN Physical Layer (DXCPL). Others Reserved. $0_H$ DXCPL will be disabled if not already activated by the tool after power-on-reset. $1_H$ DXCPL will stay enabled/activated for standard CAN pins. $2_H$ DXCPL is enabled for alternative CAN pins.
<b>RES</b>	31:16	r	<b>Reserved</b> Always read as 0; should be written with 0.

**Table 162 Reset Values of HF\_PROCONDBG**

Reset Type	Reset Value	Note
Application Reset	0000 0100 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### 6.5.3.2.10 Suspend

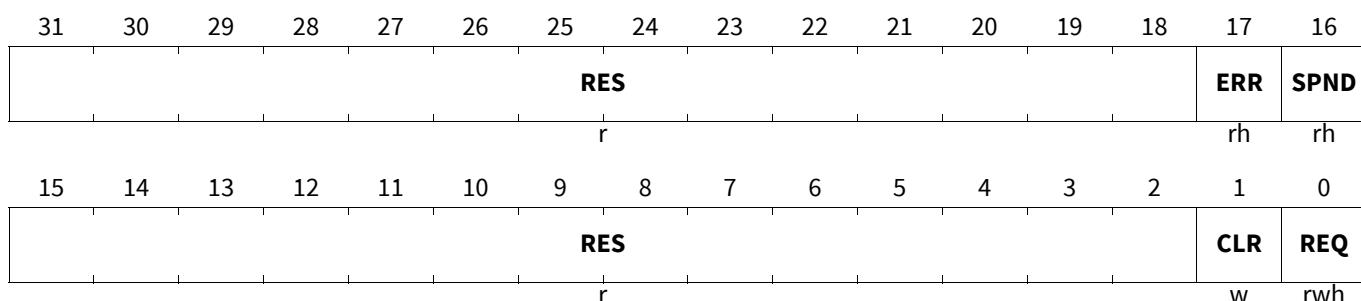
#### Suspend Control Register

##### HF\_SUSPEND

##### Suspend Control Register

(00000F0<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>REQ</b>	0	rwh	<b>Suspend Request</b> 0 <sub>B</sub> No action. 1 <sub>B</sub> Suspension of a Flash operation requested or pending.
<b>CLR</b>	1	w	<b>Suspend Clear</b> Software write only active high clear of Suspend Error. 0 <sub>B</sub> No action. 1 <sub>B</sub> Clear Suspend Error status.
<b>RES</b>	15:2, 31:18	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>SPND</b>	16	rh	<b>Flash Operation Suspended</b> Suspension of a Flash program or erase operation. 0 <sub>B</sub> No Flash operation is suspended. 1 <sub>B</sub> Flash operation is suspended.
<b>ERR</b>	17	rh	<b>Suspend Error</b>  Note: Reset by hardware when <b>HF_SUSPEND.CLR</b> is written to 1 <sub>B</sub> . 0 <sub>B</sub> No suspend error. 1 <sub>B</sub> Last suspend request via DMU_HF_SUSPEND failed.

### 6.5.3.2.11 Margin Check Control

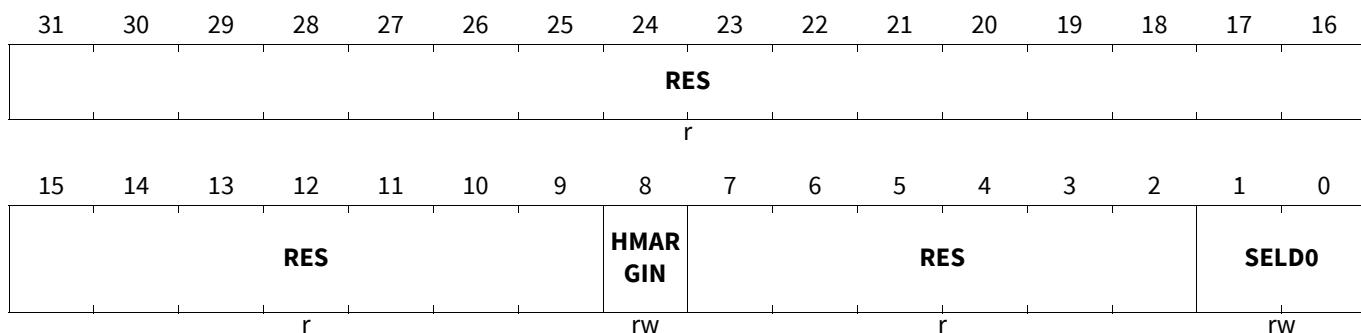
#### Margin Control Register

##### HF\_MARGIN

##### Margin Control Register

(00000F4<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>SELDO</b>	1:0	rw	<b>DF0 Margin Read Selection</b> <p><b>Note:</b> If a change between the standard and hard read margin setting is done, the system must wait a delay time <math>t_{FL\_MarginDel}</math> until the next read is requested.</p> <p>00<sub>B</sub> Read with standard margin. ... 10<sub>B</sub> Read with standard margin. 11<sub>B</sub> Read with hard margin selected by DMU_HF_MARGIN.HMARGIN.</p>
<b>RES</b>	7:2, 31:9	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>HMARGIN</b>	8	rw	<b>Hard Margin Selection</b> <p><b>Note:</b> Suboptimal 0-bits are read as 1s.</p> <p><b>Note:</b> Suboptimal 1-bits are read as 0s.</p> <p>The concrete margin values are restored from the configuration sector and are determined by Infineon.</p> <p>0<sub>B</sub> <b>Tight0</b>, Tight margin for 0 (low) level. 1<sub>B</sub> <b>Tight1</b>, Tight margin for 1 (high) level.</p>

### 6.5.3.2.12 Access Protection Registers

The master specific access control is configured and controlled by the registers **HF\_ACCEN1** and **HF\_ACCENO**.

#### Access Enable Register 0

The Access Enable Register 0 controls access for transactions with the on chip bus master TAG ID  $000000_B$  to  $011111_B$  (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

**HF\_ACCENO** provides one enable bit for each 6-bit On Chip Bus Master TAG ID encoding.

Mapping of TAG IDs to **HF\_ACCENO**.ENx: EN0 -> TAG ID  $000000_B$ , EN1 -> TAG ID  $000001_B$ , ..., EN31 -> TAG ID  $011111_B$ .

For modules connected to SRI bus the **HF\_ACCENO** register controls write accesses.

:

Note: Please see also chapter *On-Chip System Buses and Bus Bridges*, table 'On Chip Bus Master TAG Assignments'

#### HF\_ACCENO

#### Access Enable Register 0

(00000FC<sub>H</sub>)

Application Reset Value: FFFF FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

Field	Bits	Type	Description
ENy (y=0-2)	y	rw	<b>Access Enable for Master TAG ID y</b> This bit enables read / write access to the module kernel addresses for transactions with the Master TAG ID y $0_B$ Read / Write access will not be executed $1_B$ Read / Write access will be executed
EN3	3	rw	<b>Access Enable for Master TAG ID 3</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID 3 $0_B$ Write access will not be executed $1_B$ Write access will be executed
ENy (y=4-27)	y	rw	<b>Access Enable for Master TAG ID y</b> This bit enables read / write access to the module kernel addresses for transactions with the Master TAG ID y $0_B$ Read / Write access will not be executed $1_B$ Read / Write access will be executed
EN28	28	rw	<b>Access Enable for Master TAG ID 28</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID 28 $0_B$ Write access will not be executed $1_B$ Write access will be executed

Field	Bits	Type	Description
<b>EN29</b>	29	rw	<b>Access Enable for Master TAG ID 29</b> This bit enables read / write access to the module kernel addresses for transactions with the Master TAG ID 29 $0_B$ Read / Write access will not be executed $1_B$ Read / Write access will be executed
<b>EN30</b>	30	rw	<b>Access Enable for Master TAG ID 30</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID 30 $0_B$ Write access will not be executed $1_B$ Write access will be executed
<b>EN31</b>	31	rw	<b>Access Enable for Master TAG ID 31</b> This bit enables read / write access to the module kernel addresses for transactions with the Master TAG ID 30 $0_B$ Read / Write access will not be executed $1_B$ Read / Write access will be executed

### Access Enable Register 1

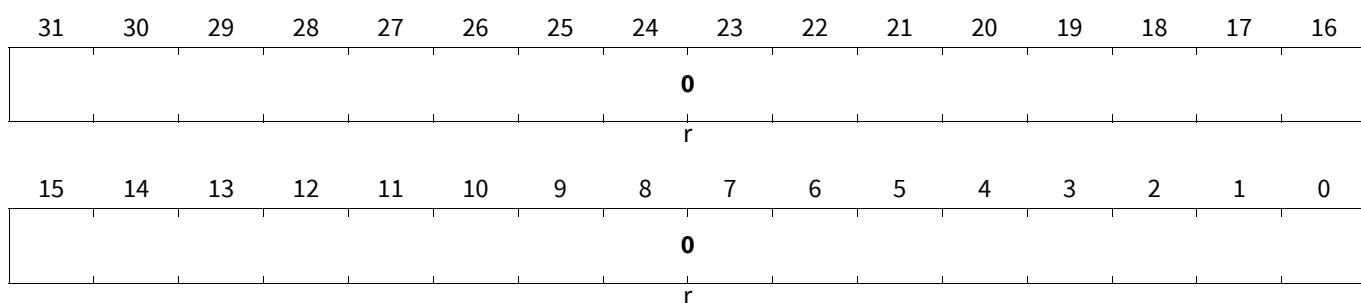
The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID  $100000_B$  to  $111111_B$  (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

**HF\_ACCE1** is not implemented with register bits as the related On Chip Bus Master TAG IDs are not used.

Mapping of TAG IDs to **HF\_ACCE1**.ENx: EN0 -> TAG ID  $100000_B$ , EN1 -> TAG ID  $100001_B$ , ..., EN31 -> TAG ID  $111111_B$ .  
Access Enable Register 1 is safe endinit protected.

#### HF\_ACCE1

**Access Enable Register 1** **(00000F8<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>0</b>	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

### 6.5.3.2.13 Protection Configuration

#### PFLASH Protection Configuration

The configuration of PFLASH read protection is indicated by the PFLASH protection configuration registers.

## PFLASH Bank i Protection Configuration 0

### HP\_PROCONPi0 (i=0-5)

PFLASH Bank i Protection Configuration 0 ( $0010000_H + i * 100_H$ )

Reset Value: [Table 163](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>S31L</b>	<b>S30L</b>	<b>S29L</b>	<b>S28L</b>	<b>S27L</b>	<b>S26L</b>	<b>S25L</b>	<b>S24L</b>	<b>S23L</b>	<b>S22L</b>	<b>S21L</b>	<b>S20L</b>	<b>S19L</b>	<b>S18L</b>	<b>S17L</b>	<b>S16L</b>
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>S15L</b>	<b>S14L</b>	<b>S13L</b>	<b>S12L</b>	<b>S11L</b>	<b>S10L</b>	<b>S9L</b>	<b>S8L</b>	<b>S7L</b>	<b>S6L</b>	<b>S5L</b>	<b>S4L</b>	<b>S3L</b>	<b>S2L</b>	<b>S1L</b>	<b>S0L</b>
rh															

Field	Bits	Type	Description
<b>SxL (x=0-31)</b>	x	rh	<b>PFLASH p Sector x Locked for Write Protection</b> These bits indicate whether PFLASH sector x is write-protected. $0_B$ No write protection is configured for sector x. $1_B$ Write protection is configured for sector x.

**Table 163 Reset Values of HP\_PROCONPi0 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

## PFLASH Bank i Protection Configuration 1

### HP\_PROCONPi1 (i=0-5)

PFLASH Bank i Protection Configuration 1 ( $0010004_H + i * 100_H$ )

Reset Value: [Table 164](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>S63L</b>	<b>S62L</b>	<b>S61L</b>	<b>S60L</b>	<b>S59L</b>	<b>S58L</b>	<b>S57L</b>	<b>S56L</b>	<b>S55L</b>	<b>S54L</b>	<b>S53L</b>	<b>S52L</b>	<b>S51L</b>	<b>S50L</b>	<b>S49L</b>	<b>S48L</b>
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>S47L</b>	<b>S46L</b>	<b>S45L</b>	<b>S44L</b>	<b>S43L</b>	<b>S42L</b>	<b>S41L</b>	<b>S40L</b>	<b>S39L</b>	<b>S38L</b>	<b>S37L</b>	<b>S36L</b>	<b>S35L</b>	<b>S34L</b>	<b>S33L</b>	<b>S32L</b>
rh															

Field	Bits	Type	Description
<b>SxL (x=32-63)</b>	x-32	rh	<b>PFLASH p Sector x Locked for Write Protection</b> These bits indicate whether PFLASH sector x is write-protected. $0_B$ No write protection is configured for sector x. $1_B$ Write protection is configured for sector x.

**Table 164 Reset Values of HP\_PROCONPi1 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### PFLASH Bank i Protection Configuration 2

#### HP\_PROCONPi2 (i=0-5)

PFLASH Bank i Protection Configuration 2 (0010008<sub>H</sub>+i\*100<sub>H</sub>)

Reset Value: [Table 165](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>S95L</b>	<b>S94L</b>	<b>S93L</b>	<b>S92L</b>	<b>S91L</b>	<b>S90L</b>	<b>S89L</b>	<b>S88L</b>	<b>S87L</b>	<b>S86L</b>	<b>S85L</b>	<b>S84L</b>	<b>S83L</b>	<b>S82L</b>	<b>S81L</b>	<b>S80L</b>
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>S79L</b>	<b>S78L</b>	<b>S77L</b>	<b>S76L</b>	<b>S75L</b>	<b>S74L</b>	<b>S73L</b>	<b>S72L</b>	<b>S71L</b>	<b>S70L</b>	<b>S69L</b>	<b>S68L</b>	<b>S67L</b>	<b>S66L</b>	<b>S65L</b>	<b>S64L</b>
rh															

Field	Bits	Type	Description
<b>SxL (x=64-95)</b>	x-64	rh	<b>PFLASH p Sector x Locked for Write Protection</b> These bits indicate whether PFLASH sector x is write-protected. 0 <sub>B</sub> No write protection is configured for sector x. 1 <sub>B</sub> Write protection is configured for sector x.

**Table 165 Reset Values of HP\_PROCONPi2 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### PFLASH Bank i Protection Configuration 3

#### HP\_PROCONPi3 (i=0-5)

PFLASH Bank i Protection Configuration 3 (001000C<sub>H</sub>+i\*100<sub>H</sub>)

Reset Value: [Table 166](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>S127L</b>	<b>S126L</b>	<b>S125L</b>	<b>S124L</b>	<b>S123L</b>	<b>S122L</b>	<b>S121L</b>	<b>S120L</b>	<b>S119L</b>	<b>S118L</b>	<b>S117L</b>	<b>S116L</b>	<b>S115L</b>	<b>S114L</b>	<b>S113L</b>	<b>S112L</b>
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>S111L</b>	<b>S110L</b>	<b>S109L</b>	<b>S108L</b>	<b>S107L</b>	<b>S106L</b>	<b>S105L</b>	<b>S104L</b>	<b>S103L</b>	<b>S102L</b>	<b>S101L</b>	<b>S100L</b>	<b>S99L</b>	<b>S98L</b>	<b>S97L</b>	<b>S96L</b>
rh															

Field	Bits	Type	Description
<b>SxL (x=96-127)</b>	x-96	rh	<b>PFLASH p Sector x Locked for Write Protection</b> These bits indicate whether PFLASH sector x is write-protected. 0 <sub>B</sub> No write protection is configured for sector x. 1 <sub>B</sub> Write protection is configured for sector x.

**Table 166 Reset Values of HP\_PROCONPi3 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

#### PFLASH Bank i Protection Configuration 4

##### HP\_PROCONPi4 (i=0-5)

##### PFLASH Bank i Protection Configuration 4 (0010010<sub>H</sub>+i\*100<sub>H</sub>)

Reset Value: [Table 167](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>S159L</b>	<b>S158L</b>	<b>S157L</b>	<b>S156L</b>	<b>S155L</b>	<b>S154L</b>	<b>S153L</b>	<b>S152L</b>	<b>S151L</b>	<b>S150L</b>	<b>S149L</b>	<b>S148L</b>	<b>S147L</b>	<b>S146L</b>	<b>S145L</b>	<b>S144L</b>
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>S143L</b>	<b>S142L</b>	<b>S141L</b>	<b>S140L</b>	<b>S139L</b>	<b>S138L</b>	<b>S137L</b>	<b>S136L</b>	<b>S135L</b>	<b>S134L</b>	<b>S133L</b>	<b>S132L</b>	<b>S131L</b>	<b>S130L</b>	<b>S129L</b>	<b>S128L</b>
rh															

Field	Bits	Type	Description
<b>SxL (x=128-159)</b>	x-128	rh	<b>PFLASH p Sector x Locked for Write Protection</b> These bits indicate whether PFLASH sector x is write-protected. 0 <sub>B</sub> No write protection is configured for sector x. 1 <sub>B</sub> Write protection is configured for sector x.

**Table 167 Reset Values of HP\_PROCONPi4 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

## PFLASH Bank i Protection Configuration 5

### HP\_PROCONPi5 (i=0-5)

PFLASH Bank i Protection Configuration 5 ( $0010014_H + i * 100_H$ )

Reset Value: [Table 168](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S191L	S190L	S189L	S188L	S187L	S186L	S185L	S184L	S183L	S182L	S181L	S180L	S179L	S178L	S177L	S176L
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S175L	S174L	S173L	S172L	S171L	S170L	S169L	S168L	S167L	S166L	S165L	S164L	S163L	S162L	S161L	S160L
rh															

Field	Bits	Type	Description
SxL (x=160-191)	x-160	rh	<b>PFLASH p Sector x Locked for Write Protection</b> These bits indicate whether PFLASH sector x is write-protected. 0 <sub>B</sub> No write protection is configured for sector x. 1 <sub>B</sub> Write protection is configured for sector x.

**Table 168 Reset Values of HP\_PROCONPi5 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

## OTP Protection Configuration

After Flash startup this register represents the or-combination of all PROCONOTP entries of all confirmed configuration sets in UCB OTP.

## PFLASH Bank i OTP Protection Configuration 0

### HP\_PROCONOTP0 (i=0-5)

PFLASH Bank i OTP Protection Configuration 0 ( $0010040_H + i * 100_H$ )

Reset Value: [Table 169](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31RO M	S30RO M	S29RO M	S28RO M	S27RO M	S26RO M	S25RO M	S24RO M	S23RO M	S22RO M	S21RO M	S20RO M	S19RO M	S18RO M	S17RO M	S16RO M
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15RO M	S14RO M	S13RO M	S12RO M	S11RO M	S10RO M	S9RO M	S8RO M	S7RO M	S6RO M	S5RO M	S4RO M	S3RO M	S2RO M	S1RO M	S0RO M
rh															

Field	Bits	Type	Description
SxROM (x=0-31)	x	rh	<p><b>PFLASH p Sector x Locked for Forever</b></p> <p>These bits indicate whether PFLASH p sector x is an OTP protected sector with read-only functionality.</p> <p>0<sub>B</sub> No OTP protection is configured for sector x. 1<sub>B</sub> OTP protection is configured for sector x.</p>

**Table 169 Reset Values of HP\_PROCONOTPi0 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**PFLASH Bank i OTP Protection Configuration 1****HP\_PROCONOTPi1 (i=0-5)****PFLASH Bank i OTP Protection Configuration 1(0010044<sub>H</sub>+i\*100<sub>H</sub>)****Reset Value: Table 170**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63RO M	S62RO M	S61RO M	S60RO M	S59RO M	S58RO M	S57RO M	S56RO M	S55RO M	S54RO M	S53RO M	S52RO M	S51RO M	S50RO M	S49RO M	S48RO M
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47RO M	S46RO M	S45RO M	S44RO M	S43RO M	S42RO M	S41RO M	S40RO M	S39RO M	S38RO M	S37RO M	S36RO M	S35RO M	S34RO M	S33RO M	S32RO M
rh															

Field	Bits	Type	Description
SxROM (x=32-63)	x-32	rh	<p><b>PFLASH p Sector x Locked for Forever</b></p> <p>These bits indicate whether PFLASH p sector x is an OTP protected sector with read-only functionality.</p> <p>0<sub>B</sub> No OTP protection is configured for sector x. 1<sub>B</sub> OTP protection is configured for sector x.</p>

**Table 170 Reset Values of HP\_PROCONOTPi1 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

## PFLASH Bank i OTP Protection Configuration 2

### HP\_PROCONOTPi2 (i=0-5)

PFLASH Bank i OTP Protection Configuration 2( $0010048_H + i * 100_H$ )

Reset Value: [Table 171](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95RO M	S94RO M	S93RO M	S92RO M	S91RO M	S90RO M	S89RO M	S88RO M	S87RO M	S86RO M	S85RO M	S84RO M	S83RO M	S82RO M	S81RO M	S80RO M
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79RO M	S78RO M	S77RO M	S76RO M	S75RO M	S74RO M	S73RO M	S72RO M	S71RO M	S70RO M	S69RO M	S68RO M	S67RO M	S66RO M	S65RO M	S64RO M
rh															

Field	Bits	Type	Description
SxROM (x=64-95)	x-64	rh	<b>PFLASH p Sector x Locked for Forever</b> These bits indicate whether PFLASH p sector x is an OTP protected sector with read-only functionality. 0 <sub>B</sub> No OTP protection is configured for sector x. 1 <sub>B</sub> OTP protection is configured for sector x.

**Table 171 Reset Values of HP\_PROCONOTPi2 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

## PFLASH Bank i OTP Protection Configuration 3

### HP\_PROCONOTPi3 (i=0-5)

PFLASH Bank i OTP Protection Configuration 3( $001004C_H + i * 100_H$ )

Reset Value: [Table 172](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127R OM	S126R OM	S125R OM	S124R OM	S123R OM	S122R OM	S121R OM	S120R OM	S119R OM	S118R OM	S117R OM	S116R OM	S115R OM	S114R OM	S113R OM	S112R OM
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111R OM	S110R OM	S109R OM	S108R OM	S107R OM	S106R OM	S105R OM	S104R OM	S103R OM	S102R OM	S101R OM	S100R OM	S99RO M	S98RO M	S97RO M	S96RO M
rh															

Field	Bits	Type	Description
SxROM (x=96-127)	x-96	rh	<p><b>PFLASH p Sector x Locked for Forever</b></p> <p>These bits indicate whether PFLASH p sector x is an OTP protected sector with read-only functionality.</p> <p>0<sub>B</sub> No OTP protection is configured for sector x. 1<sub>B</sub> OTP protection is configured for sector x.</p>

**Table 172 Reset Values of HP\_PROCONOTPi3 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**PFLASH Bank i OTP Protection Configuration 4****HP\_PROCONOTPi4 (i=0-5)****PFLASH Bank i OTP Protection Configuration 4(0010050<sub>H</sub>+i\*100<sub>H</sub>)****Reset Value: Table 173**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S159R OM	S158R OM	S157R OM	S156R OM	S155R OM	S154R OM	S153R OM	S152R OM	S151R OM	S150R OM	S149R OM	S148R OM	S147R OM	S146R OM	S145R OM	S144R OM
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S143R OM	S142R OM	S141R OM	S140R OM	S139R OM	S138R OM	S137R OM	S136R OM	S135R OM	S134R OM	S133R OM	S132R OM	S131R OM	S130R OM	S129R OM	S128R OM
rh															

Field	Bits	Type	Description
SxROM (x=128-159)	x-128	rh	<p><b>PFLASH p Sector x Locked for Forever</b></p> <p>These bits indicate whether PFLASH p sector x is an OTP protected sector with read-only functionality.</p> <p>0<sub>B</sub> No OTP protection is configured for sector x. 1<sub>B</sub> OTP protection is configured for sector x.</p>

**Table 173 Reset Values of HP\_PROCONOTPi4 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### PFLASH Bank i OTP Protection Configuration 5

#### HP\_PROCONOTPi5 (i=0-5)

PFLASH Bank i OTP Protection Configuration 5( $0010054_H+i*100_H$ )

Reset Value: [Table 174](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S191R OM	S190R OM	S189R OM	S188R OM	S187R OM	S186R OM	S185R OM	S184R OM	S183R OM	S182R OM	S181R OM	S180R OM	S179R OM	S178R OM	S177R OM	S176R OM
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S175R OM	S174R OM	S173R OM	S172R OM	S171R OM	S170R OM	S169R OM	S168R OM	S167R OM	S166R OM	S165R OM	S164R OM	S163R OM	S162R OM	S161R OM	S160R OM
rh															

Field	Bits	Type	Description
SxROM (x=160-191)	x-160	rh	<b>PFLASH p Sector x Locked for Forever</b> These bits indicate whether PFLASH p sector x is an OTP protected sector with read-only functionality. 0 <sub>B</sub> No OTP protection is configured for sector x. 1 <sub>B</sub> OTP protection is configured for sector x.

**Table 174 Reset Values of HP\_PROCONOTPi5 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### Write-Once Protection Configuration

After Flash startup this register represents the or-combination of all PROCONWOP entries of all confirmed configuration sets in UCB\_OTP.

### PFLASH Bank i WOP Configuration 0

#### HP\_PROCONWOPi0 (i=0-5)

PFLASH Bank i WOP Configuration 0

( $0010080_H+i*100_H$ )

Reset Value: [Table 175](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31W OP	S30W OP	S29W OP	S28W OP	S27W OP	S26W OP	S25W OP	S24W OP	S23W OP	S22W OP	S21W OP	S20W OP	S19W OP	S18W OP	S17W OP	S16W OP
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15W OP	S14W OP	S13W OP	S12W OP	S11W OP	S10W OP	S9WO P	S8WO P	S7WO P	S6WO P	S5WO P	S4WO P	S3WO P	S2WO P	S1WO P	S0WO P
rh															

Field	Bits	Type	Description
SxWOP (x=0-31)	x	rh	<b>PFLASH p Sector x Configured for Write-Once Protection</b> These bits indicate whether PFLASH p sector x is an WOP protected sector. 0 <sub>B</sub> No WOP protection is configured for sector x. 1 <sub>B</sub> WOP protection is configured for sector x.

**Table 175 Reset Values of HP\_PROCONWOPi0 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### PFLASH Bank i WOP Configuration 1

#### HP\_PROCONWOPi1 (i=0-5)

PFLASH Bank i WOP Configuration 1 (0010084<sub>H</sub>+i\*100<sub>H</sub>)

Reset Value: [Table 176](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63W OP	S62W OP	S61W OP	S60W OP	S59W OP	S58W OP	S57W OP	S56W OP	S55W OP	S54W OP	S53W OP	S52W OP	S51W OP	S50W OP	S49W OP	S48W OP
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47W OP	S46W OP	S45W OP	S44W OP	S43W OP	S42W OP	S41W OP	S40W OP	S39W OP	S38W OP	S37W OP	S36W OP	S35W OP	S34W OP	S33W OP	S32W OP
rh															

Field	Bits	Type	Description
SxWOP (x=32-63)	x-32	rh	<b>PFLASH p Sector x Configured for Write-Once Protection</b> These bits indicate whether PFLASH p sector x is an WOP protected sector. 0 <sub>B</sub> No WOP protection is configured for sector x. 1 <sub>B</sub> WOP protection is configured for sector x.

**Table 176 Reset Values of HP\_PROCONWOPi1 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### PFLASH Bank i WOP Configuration 2

#### HP\_PROCONWOPi2 (i=0-5)

##### PFLASH Bank i WOP Configuration 2

(0010088<sub>H</sub>+i\*100<sub>H</sub>)

Reset Value: [Table 177](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95W OP	S94W OP	S93W OP	S92W OP	S91W OP	S90W OP	S89W OP	S88W OP	S87W OP	S86W OP	S85W OP	S84W OP	S83W OP	S82W OP	S81W OP	S80W OP
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79W OP	S78W OP	S77W OP	S76W OP	S75W OP	S74W OP	S73W OP	S72W OP	S71W OP	S70W OP	S69W OP	S68W OP	S67W OP	S66W OP	S65W OP	S64W OP
rh															

Field	Bits	Type	Description
SxWOP (x=64-95)	x-64	rh	<b>PFLASH p Sector x Configured for Write-Once Protection</b> These bits indicate whether PFLASH p sector x is an WOP protected sector. 0 <sub>B</sub> No WOP protection is configured for sector x. 1 <sub>B</sub> WOP protection is configured for sector x.

**Table 177 Reset Values of HP\_PROCONWOPi2 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### PFLASH Bank i WOP Configuration 3

#### HP\_PROCONWOPi3 (i=0-5)

##### PFLASH Bank i WOP Configuration 3

(001008C<sub>H</sub>+i\*100<sub>H</sub>)

Reset Value: [Table 178](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127W OP	S126W OP	S125W OP	S124W OP	S123W OP	S122W OP	S121W OP	S120W OP	S119W OP	S118W OP	S117W OP	S116W OP	S115W OP	S114W OP	S113W OP	S112W OP
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111W OP	S110W OP	S109W OP	S108W OP	S107W OP	S106W OP	S105W OP	S104W OP	S103W OP	S102W OP	S101W OP	S100W OP	S99W OP	S98W OP	S97W OP	S96W OP
rh															

Field	Bits	Type	Description
SxWOP (x=96-127)	x-96	rh	<p><b>PFLASH p Sector x Configured for Write-Once Protection</b></p> <p>These bits indicate whether PFLASH p sector x is an WOP protected sector.</p> <p>0<sub>B</sub> No WOP protection is configured for sector x. 1<sub>B</sub> WOP protection is configured for sector x.</p>

**Table 178 Reset Values of HP\_PROCONWOPi3 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**PFLASH Bank i WOP Configuration 4****HP\_PROCONWOPi4 (i=0-5)**

**PFLASH Bank i WOP Configuration 4 (0010090<sub>H</sub>+i\*100<sub>H</sub>)**      **Reset Value: Table 179**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S159W OP	S158W OP	S157W OP	S156W OP	S155W OP	S154W OP	S153W OP	S152W OP	S151W OP	S150W OP	S149W OP	S148W OP	S147W OP	S146W OP	S145W OP	S144W OP
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S143W OP	S142W OP	S141W OP	S140W OP	S139W OP	S138W OP	S137W OP	S136W OP	S135W OP	S134W OP	S133W OP	S132W OP	S131W OP	S130W OP	S129W OP	S128W OP
rh															

Field	Bits	Type	Description
SxWOP (x=128-159)	x-128	rh	<p><b>PFLASH p Sector x Configured for Write-Once Protection</b></p> <p>These bits indicate whether PFLASH p sector x is an WOP protected sector.</p> <p>0<sub>B</sub> No WOP protection is configured for sector x. 1<sub>B</sub> WOP protection is configured for sector x.</p>

**Table 179 Reset Values of HP\_PROCONWOPi4 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**PFLASH Bank i WOP Configuration 5****HP\_PROCONWOPi5 (i=0-5)****PFLASH Bank i WOP Configuration 5**(0010094<sub>H</sub>+i\*100<sub>H</sub>)Reset Value: [Table 180](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S191W OP	S190W OP	S189W OP	S188W OP	S187W OP	S186W OP	S185W OP	S184W OP	S183W OP	S182W OP	S181W OP	S180W OP	S179W OP	S178W OP	S177W OP	S176W OP
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S175W OP	S174W OP	S173W OP	S172W OP	S171W OP	S170W OP	S169W OP	S168W OP	S167W OP	S166W OP	S165W OP	S164W OP	S163W OP	S162W OP	S161W OP	S160W OP
rh															

Field	Bits	Type	Description
SxWOP (x=160-191)	x-160	rh	<b>PFLASH p Sector x Configured for Write-Once Protection</b> These bits indicate whether PFLASH p sector x is an WOP protected sector. 0 <sub>B</sub> No WOP protection is configured for sector x. 1 <sub>B</sub> WOP protection is configured for sector x.

**Table 180 Reset Values of HP\_PROCONWOPi5 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**Erase Counter Priority Configuration**

After Flash startup this register represents the priority configuration of the PFlash logical sectors for Erase Counter recording.

**PFLASH Bank i Erase Counter Priority configuration 0****HP\_ECPRI0i0 (i=0-5)****PFLASH Bank i Erase Counter Priority configuration 0(00100A0<sub>H</sub>+i\*100<sub>H</sub>)**Reset Value: [Table 181](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31L	S30L	S29L	S28L	S27L	S26L	S25L	S24L	S23L	S22L	S21L	S20L	S19L	S18L	S17L	S16L
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15L	S14L	S13L	S12L	S11L	S10L	S9L	S8L	S7L	S6L	S5L	S4L	S3L	S2L	S1L	S0L
rh															

Field	Bits	Type	Description
SxL (x=0-31)	x	rh	<b>PFLASH p Sector x Erase Counter priority</b> These bits indicate whether PFLASH sector x has high or low priority for Erase Counter recording. 0 <sub>B</sub> Low priority is configured for sector x. 1 <sub>B</sub> High priority is configured for sector x.

**Table 181 Reset Values of HP\_ECPRIo10 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**PFLASH Bank i Erase Counter Priority Configuration 1****HP\_ECPRIo1 (i=0-5)****PFLASH Bank i Erase Counter Priority Configuration 1(00100A4<sub>H</sub>+i\*100<sub>H</sub>)****Reset Value: Table 182**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>S63L</b>	<b>S62L</b>	<b>S61L</b>	<b>S60L</b>	<b>S59L</b>	<b>S58L</b>	<b>S57L</b>	<b>S56L</b>	<b>S55L</b>	<b>S54L</b>	<b>S53L</b>	<b>S52L</b>	<b>S51L</b>	<b>S50L</b>	<b>S49L</b>	<b>S48L</b>
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>S47L</b>	<b>S46L</b>	<b>S45L</b>	<b>S44L</b>	<b>S43L</b>	<b>S42L</b>	<b>S41L</b>	<b>S40L</b>	<b>S39L</b>	<b>S38L</b>	<b>S37L</b>	<b>S36L</b>	<b>S35L</b>	<b>S34L</b>	<b>S33L</b>	<b>S32L</b>
rh															

Field	Bits	Type	Description
SxL (x=32-63)	x-32	rh	<b>PFLASH p Sector x Erase Counter priority</b> These bits indicate whether PFLASH sector x has high or low priority for Erase Counter recording. 0 <sub>B</sub> Low priority is configured for sector x. 1 <sub>B</sub> High priority is configured for sector x.

**Table 182 Reset Values of HP\_ECPRIo1 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

## PFLASH Bank i Erase Counter Priority Configuration 2

### HP\_ECPRI0i2 (i=0-5)

PFLASH Bank i Erase Counter Priority Configuration 2(00100A8<sub>H</sub>+i\*100<sub>H</sub>)

Reset Value: [Table 183](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95L	S94L	S93L	S92L	S91L	S90L	S89L	S88L	S87L	S86L	S85L	S84L	S83L	S82L	S81L	S80L
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79L	S78L	S77L	S76L	S75L	S74L	S73L	S72L	S71L	S70L	S69L	S68L	S67L	S66L	S65L	S64L
rh															

Field	Bits	Type	Description
SxL (x=64-95)	x-64	rh	<b>PFLASH p Sector x Erase Counter priority</b> These bits indicate whether PFLASH sector x has high or low priority for Erase Counter recording. 0 <sub>B</sub> Low priority is configured for sector x. 1 <sub>B</sub> High priority is configured for sector x.

**Table 183 Reset Values of HP\_ECPRI0i2 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

## PFLASH Bank i Erase Counter Priority Configuration 3

### HP\_ECPRI0i3 (i=0-5)

PFLASH Bank i Erase Counter Priority Configuration 3(00100AC<sub>H</sub>+i\*100<sub>H</sub>)

Reset Value: [Table 184](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127L	S126L	S125L	S124L	S123L	S122L	S121L	S120L	S119L	S118L	S117L	S116L	S115L	S114L	S113L	S112L
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111L	S110L	S109L	S108L	S107L	S106L	S105L	S104L	S103L	S102L	S101L	S100L	S99L	S98L	S97L	S96L
rh															

Field	Bits	Type	Description
SxL (x=96-127)	x-96	rh	<b>PFLASH p Sector x Erase Counter priority</b> These bits indicate whether PFLASH sector x has high or low priority for Erase Counter recording. 0 <sub>B</sub> Low priority is configured for sector x. 1 <sub>B</sub> High priority is configured for sector x.

**Table 184 Reset Values of HP\_ECPRI0i3 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**PFLASH Bank i Erase Counter Priority Configuration 4****HP\_ECPRI0i4 (i=0-5)****PFLASH Bank i Erase Counter Priority Configuration 4(00100B0<sub>H</sub>+i\*100<sub>H</sub>)****Reset Value: Table 185**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>S159L</b>	<b>S158L</b>	<b>S157L</b>	<b>S156L</b>	<b>S155L</b>	<b>S154L</b>	<b>S153L</b>	<b>S152L</b>	<b>S151L</b>	<b>S150L</b>	<b>S149L</b>	<b>S148L</b>	<b>S147L</b>	<b>S146L</b>	<b>S145L</b>	<b>S144L</b>
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>S143L</b>	<b>S142L</b>	<b>S141L</b>	<b>S140L</b>	<b>S139L</b>	<b>S138L</b>	<b>S137L</b>	<b>S136L</b>	<b>S135L</b>	<b>S134L</b>	<b>S133L</b>	<b>S132L</b>	<b>S131L</b>	<b>S130L</b>	<b>S129L</b>	<b>S128L</b>
rh															

Field	Bits	Type	Description
<b>SxL (x=128-159)</b>	x-128	rh	<b>PFLASH p Sector x Erase Counter priority</b> These bits indicate whether PFLASH sector x has high or low priority for Erase Counter recording. 0 <sub>B</sub> Low priority is configured for sector x. 1 <sub>B</sub> High priority is configured for sector x.

**Table 185 Reset Values of HP\_ECPRI0i4 (i=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**PFLASH Bank i Erase Counter Priority Configuration 5****HP\_ECPRI0i5 (i=0-5)****PFLASH Bank i Erase Counter Priority Configuration 5(00100B4<sub>H</sub>+i\*100<sub>H</sub>)****Reset Value: Table 186**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>S191L</b>	<b>S190L</b>	<b>S189L</b>	<b>S188L</b>	<b>S187L</b>	<b>S186L</b>	<b>S185L</b>	<b>S184L</b>	<b>S183L</b>	<b>S182L</b>	<b>S181L</b>	<b>S180L</b>	<b>S179L</b>	<b>S178L</b>	<b>S177L</b>	<b>S176L</b>
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>S175L</b>	<b>S174L</b>	<b>S173L</b>	<b>S172L</b>	<b>S171L</b>	<b>S170L</b>	<b>S169L</b>	<b>S168L</b>	<b>S167L</b>	<b>S166L</b>	<b>S165L</b>	<b>S164L</b>	<b>S163L</b>	<b>S162L</b>	<b>S161L</b>	<b>S160L</b>
rh															

Field	Bits	Type	Description
SxL (x=160-191)	x-160	rh	<b>PFLASH p Sector x Erase Counter priority</b> These bits indicate whether PFLASH sector x has high or low priority for Erase Counter recording. 0 <sub>B</sub> Low priority is configured for sector x. 1 <sub>B</sub> High priority is configured for sector x.

Table 186 Reset Values of HP\_ECPRIo5 (i=0-5)

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### 6.5.3.2.14 HSM Command Interface

The following registers constitute the HSM command interface together with its reserved address range for command sequences.

#### HSM Flash Status Register

Note: *The DxBUSY flags cannot be cleared with the “Clear Status” command or with the “Reset to Read” command. These flags are controlled by HW.*

Note: *After every reset, the busy bits are set while the Flash module is busy with startup (until operation mode is entered). Also the protection installation bits are always set until end of startup.*

#### SF\_STATUS

HSM Flash Status Register (0020010 <sub>H</sub> ) Application Reset Value: 0000 0002 <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES										DFPAG E	RES				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES												D1BU SY	RES		

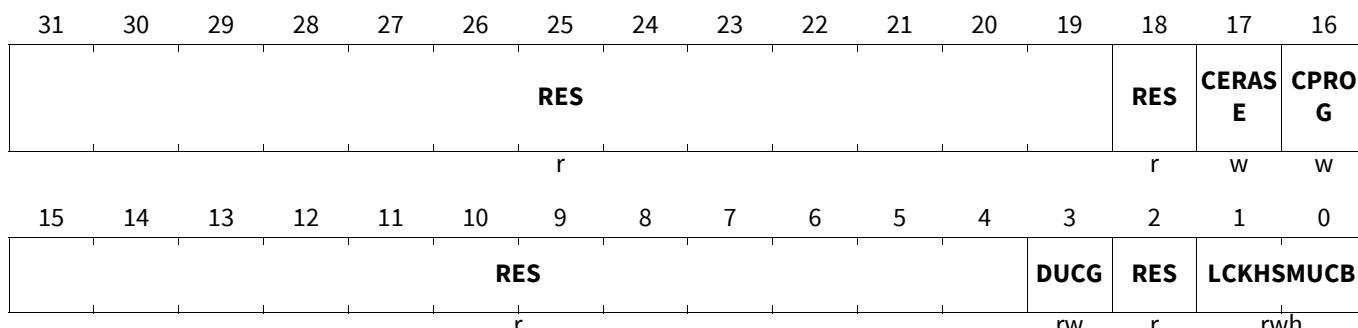
Field	Bits	Type	Description
RES	0, 19:2, 31:21	r	<b>Reserved</b> Always read as 0; should be written with 0.

Field	Bits	Type	Description
D1BUSY	1	rh	<p><b>Data Flash Bank 1 Busy</b>            HW-controlled status flag.            Indication of busy state of DFLASH bank 1 because of active execution of an operation; DF1 busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state the DF1 does not allow read access.</p> <p>0<sub>B</sub> DF1 ready, not busy; DF1 in operation mode.             1<sub>B</sub> DF1 busy; DF1 not in operation mode.</p>
DFPAGE	20	rh	<p><b>Data Flash in Page Mode</b>            HW-controlled status flag.            Set with Enter Page Mode for DFLASH, cleared with Write Page command.</p> <p><i>Note:</i> <i>Read accesses are allowed while in page mode.</i></p> <p>0<sub>B</sub> Data Flash not in page mode             1<sub>B</sub> Data Flash in page mode</p>

### HSM Flash Configuration Register

#### SF\_CONTROL

#### HSM Flash Configuration Register

(0020014<sub>H</sub>)Application Reset Value: 0000 0001<sub>H</sub>

Field	Bits	Type	Description
LCKHSMUCB	1:0	rwh	<p><b>Lock Access to UCB_HSMCFG</b>            Trap door register. This field can only be written to the “Locked” state. Other writes are ignored.</p> <p>00<sub>B</sub> <b>Locked</b>, Reads to UCB_HSMCFG forbidden.             01<sub>B</sub> <b>Unlocked</b>, Reads by HSM to UCB_HSMCFG allowed.             10<sub>B</sub> <b>Locked</b>, Reads to UCB_HSMCFG forbidden.             11<sub>B</sub> <b>Locked</b>, Reads to UCB_HSMCFG forbidden.</p>
RES	2, 15:4, 18, 31:19	r	<b>Reserved</b>

Field	Bits	Type	Description
<b>DUCG</b>	3	rw	<b>DFLASH User Command Granularity</b> Granularity configuration of User commands for DFLASH1 $0_B$ Wordline granularity. $1_B$ Page granularity.
<b>CPROG</b>	16	w	<b>Clear Programming State</b> $0_B$ No action. $1_B$ Clear the Programming State Flag DMU_SF_OPERATION.PROG
<b>CERASE</b>	17	w	<b>Clear Erase State</b> $0_B$ No action. $1_B$ Clear the Erase State Flag DMU_SF_OPERATION.ERASE

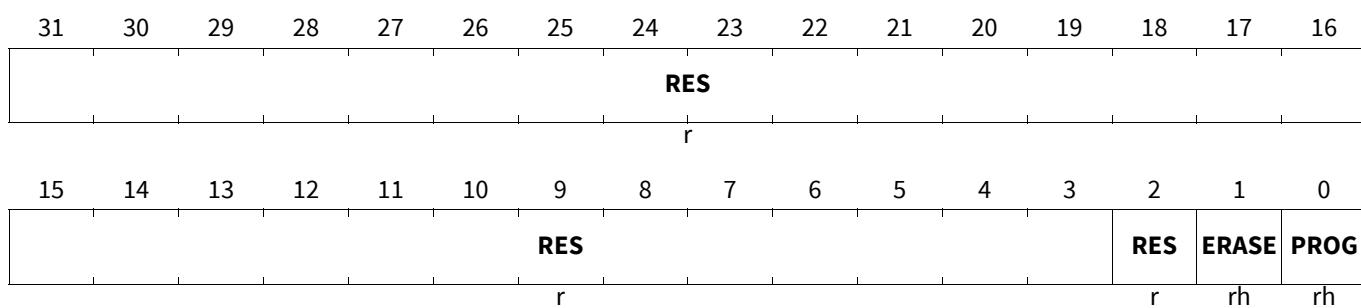
### HSM Flash Operation Register

#### SF\_OPERATION

##### HSM Flash Operation Register

(0020018<sub>H</sub>)

System Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>PROG</b>	0	rh	<b>Programming State</b> HW-controlled status flag. Set with last cycle of Write Page/Burst command sequence. If one BUSY flag is coincidentally set, PROG indicates the type of busy state. If OPER is coincidentally set, PROG indicates the type of erroneous operation. Otherwise, PROG indicates, that operation is still requested or finished. Can be also cleared by writing '11' to <b>SF_CONTROL.CPROG</b> . This bit is not set for by program operations initiated by the Host interface.  <i>Note:</i> Cleared with command "Clear Status".  $0_B$ There is no program operation requested or in progress or just finished. $1_B$ Programming operation requested or in action or finished.

Field	Bits	Type	Description
<b>ERASE</b>	1	rh	<p><b>Erase State</b>            HW-controlled status flag.            Set with last cycle of Erase/Verify command sequence.            Indications are analogous to PROG flag.            Can be also cleared by writing '11' to <b>SF_CONTROL.CERASE</b>.            This bit is not set for by erase operations initiated by the Host interface.</p> <p><i>Note:</i> Cleared with command "Clear Status".</p> <p>0<sub>B</sub> There is no erase operation requested or in progress or just finished            1<sub>B</sub> Erase/Verify operation requested or in action or finished.</p>
<b>RES</b>	2, 31:3	r	<b>Reserved</b>

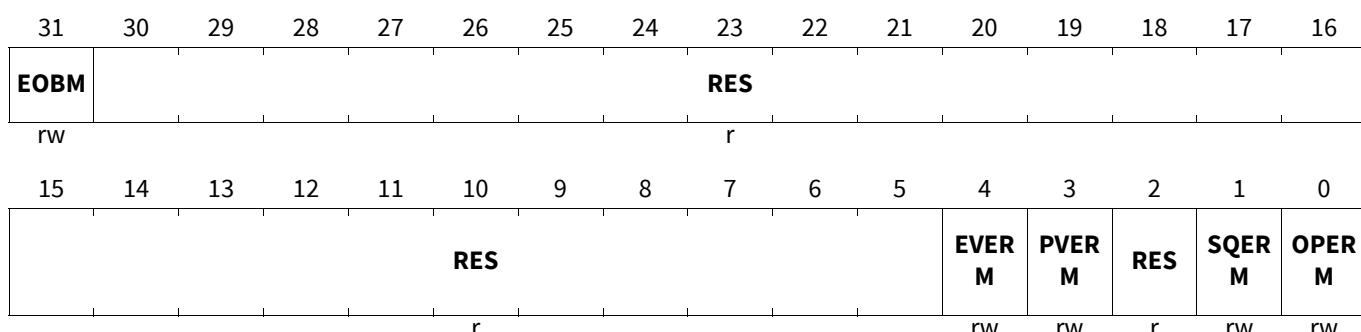
### 6.5.3.2.15 HSM Flash Error Registers

#### HSM Enable Error Interrupt Control Register

##### SF\_EER

HSM Enable Error Interrupt Control Register (0020030<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



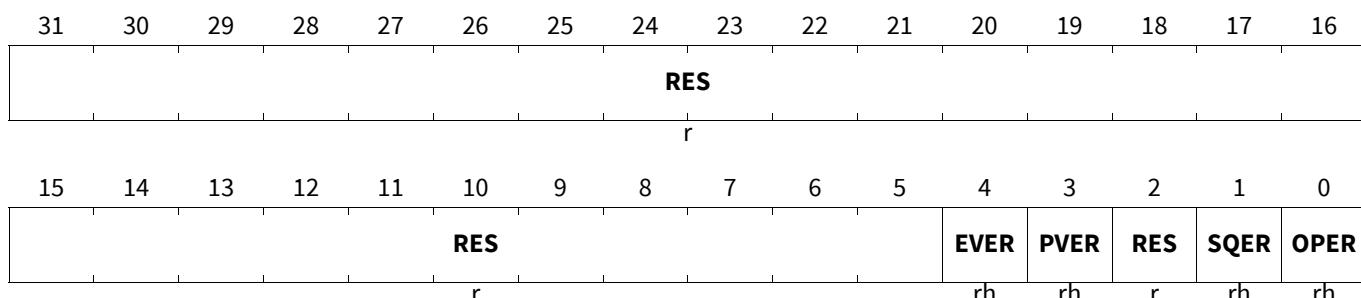
Field	Bits	Type	Description
<b>OPERM</b>	0	rw	<p><b>Operation Error Interrupt Mask</b>            0<sub>B</sub> Interrupt disabled.            1<sub>B</sub> Flash interrupt because Operation Error is enabled.</p>
<b>SQERM</b>	1	rw	<p><b>Command Sequence Error Interrupt Mask</b>            0<sub>B</sub> Interrupt disabled.            1<sub>B</sub> Flash interrupt because Sequence Error is enabled.</p>
<b>RES</b>	2, 30:5	r	<p><b>Reserved</b>            Always read as 0; should be written with 0.</p>
<b>PVERM</b>	3	rw	<p><b>Program Verify Error Interrupt Mask</b>            0<sub>B</sub> Interrupt disabled.            1<sub>B</sub> Flash interrupt because Program Verify Error is enabled.</p>
<b>EVERM</b>	4	rw	<p><b>Erase Verify Error Interrupt Mask</b>            0<sub>B</sub> Interrupt disabled.            1<sub>B</sub> Flash interrupt because Erase Verify Error is enabled.</p>

Field	Bits	Type	Description
<b>EOBM</b>	31	rw	<b>End of Busy Interrupt Mask</b> 0 <sub>B</sub> Interrupt disabled. 1 <sub>B</sub> EOB interrupt is enabled.

### HSM Error Status Register

#### SF\_ERRSR

##### HSM Error Status Register

(0020034<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

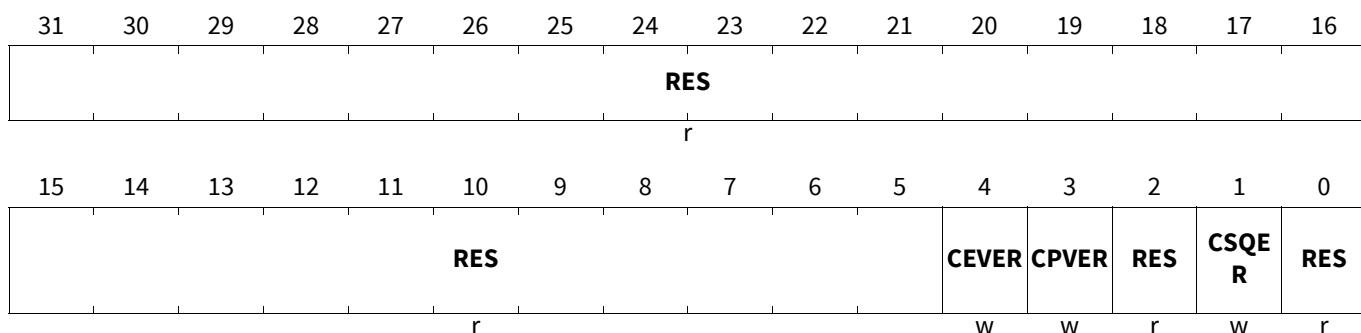
Field	Bits	Type	Description
<b>OPER</b>	0	rh	<b>Flash Operation Error</b>  <i>Note:</i> Cleared with system reset.  0 <sub>B</sub> No operation error. 1 <sub>B</sub> Flash array operation aborted, because of a Flash array failure, e.g. an ECC error in microcode.
<b>SQER</b>	1	rh	<b>Command Sequence Error</b> A sequence error is not indicated if the Reset to Read command aborts a command sequence.  <i>Note:</i> Cleared with application reset, commands “Reset to Read” and “Clear Status” or writing <b>SF_CLRE.CSQER = 1<sub>B</sub></b> .  0 <sub>B</sub> No sequence error 1 <sub>B</sub> Command state machine operation unsuccessful because of improper address or command sequence.
<b>RES</b>	2, 31:5	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>PVER</b>	3	rh	<b>Program Verify Error</b> A verify error was reported on completion of a Flash program operation.  <i>Note:</i> Cleared with application reset, with the command “Clear Status” or writing <b>SF_CLRE.CPVER = 1<sub>B</sub></b> .  0 <sub>B</sub> The page is correctly programmed. All bits have full expected quality. 1 <sub>B</sub> A program verify error has been detected. Full quality of all bits cannot be guaranteed.

Field	Bits	Type	Description
EVER	4	rh	<p><b>Erase Verify Error</b> A verify error was reported on completion of a Flash erase operation.</p> <p><i>Note:</i> Cleared with application reset, with the command “Clear Status” or writing <b>SF_CLRE.CEVER = 1<sub>B</sub></b>.</p> <p>0<sub>B</sub> The sector is correctly erased. All erased bits have full expected quality. 1<sub>B</sub> An erase verify error has been detected. Full quality erased bits cannot be guaranteed.</p>

### HSM Clear Error Register

#### SF\_CLRE

#### HSM Clear Error Register

(00200038<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
RES	0, 2, 31:5	r	<p><b>Reserved</b> Always read as 0; should be written with 0.</p>
CSQER	1	w	<p><b>Clear Command Sequence Error</b> 0<sub>B</sub> No action 1<sub>B</sub> Clear Command Sequence Error Flag DMU_SF_ERRSR.SQER</p>
CPVER	3	w	<p><b>Clear Program Verify Error</b> 0<sub>B</sub> No action 1<sub>B</sub> Clear Program Verify Error Flag DMU_SF_ERRSR.PVER</p>
CEVER	4	w	<p><b>Clear Erase Verify Error</b> 0<sub>B</sub> No action 1<sub>B</sub> Clear Erase Verify Error Flag DMU_SF_ERRSR.EVER</p>

### 6.5.3.2.16 Data Flash Bank 1 ECC Registers

#### HSM DF1 ECC Read Register

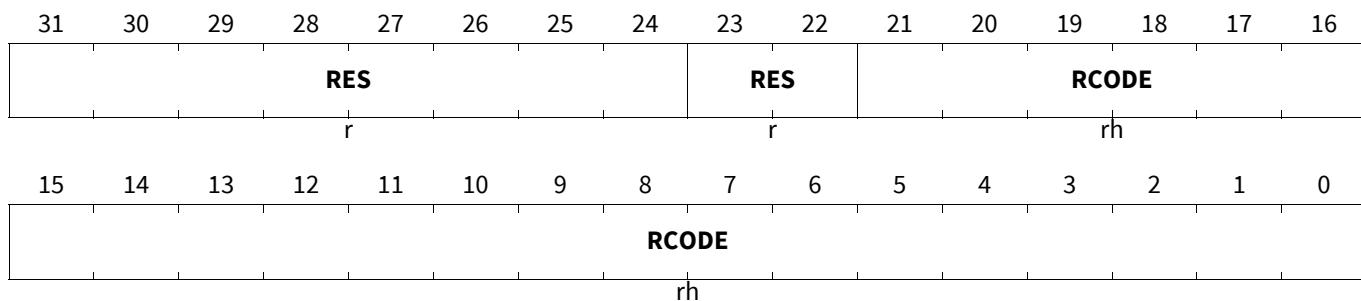
If DF1 is configured as HSM\_exclusive, then the ECC Read Register stores the ECC checksum read during the last DF1 NVM read access.

### SF\_ECCR

#### HSM DF1 ECC Read Register

(0020040<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>RCODE</b>	21:0	rh	<b>Error Correction Read Code</b> ECC checksum read during the last NVM read access.
<b>RES</b>	23:22, 31:24	r	<b>Reserved</b>

### HSM DF1 ECC Status Register

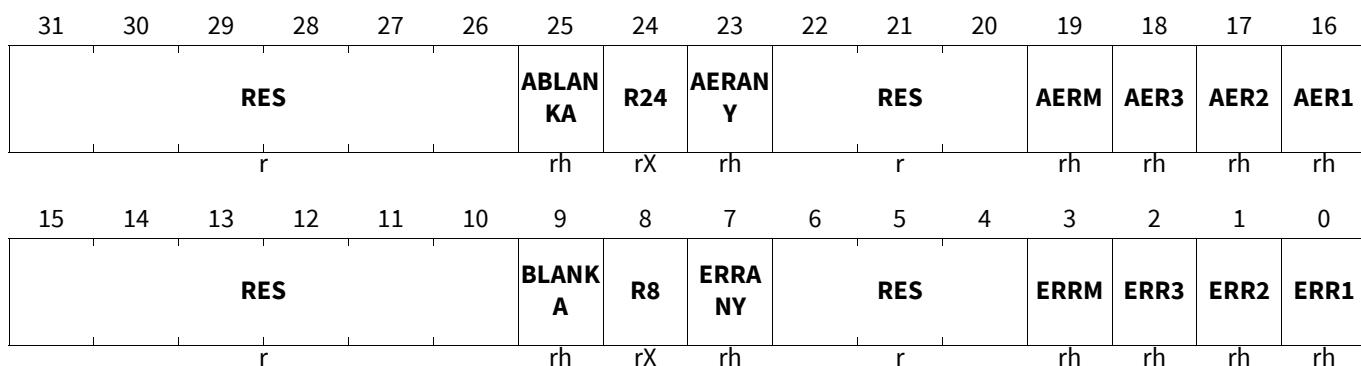
If DF1 is configured as HSM\_exclusive, then the ECC Status Register captures the ECC errors detected during the last DF1 NVM read access.

### SF\_ECCS

#### HSM DF1 ECC Status Register

(0020044<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>ERR1</b>	0	rh	<b>Read Access Single Bit ECC Error</b> The flag reports a single bit ECC failure during the last NVM read access.  Note: Reset by hardware when <b>SF_ECCC.CLR</b> is written to 11 <sub>B</sub> .  0 <sub>B</sub> No single bit ECC failure occurred. 1 <sub>B</sub> A single bit ECC failure occurred.

Field	Bits	Type	Description
ERR2	1	rh	<p><b>Read Access Double Bit ECC Error</b>            The flag reports a double bit ECC failure during the last NVM read access.</p> <p><i>Note:</i> Reset by hardware when <b>SF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> No double bit ECC failure occurred.  <math>1_B</math> A double bit ECC failure occurred.</p>
ERR3	2	rh	<p><b>Read Access Triple Bit ECC Error</b>            The flag reports a triple bit ECC failure during the last NVM read access.</p> <p><i>Note:</i> Reset by hardware when <b>SF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> No triple bit ECC failure occurred.  <math>1_B</math> A triple bit ECC failure occurred.</p>
ERRM	3	rh	<p><b>Read Access Multi-bit ECC Error</b>            The flag reports multi bit ECC failure during the last NVM read access.</p> <p><i>Note:</i> Reset by hardware when <b>SF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> No multi bit ECC failure occurred.  <math>1_B</math> Multi bit ECC failure occurred.</p>
RES	6:4, 15:10, 22:20, 31:26	r	<p><b>Reserved</b>            Always read as 0; should be written with 0.</p>
ERRANY	7	rh	<p><b>Any Read Access ECC Error</b>            The flag reports any ECC failure during the last NVM read access.</p> <p><i>Note:</i> Reset by hardware when <b>SF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> No ECC failure occurred.  <math>1_B</math> ECC failure occurred.</p>
R8	8	rX	<p><b>Reserved - RES</b></p>
BLANKA	9	rh	<p><b>Read Access Blank Analog</b>            The flag reports that all read data cells have sufficient high current: a program of new data without prior erase is possible. Under certain operation history, a valid complement data entry may also appear as blank. Data qualifiers like headers or footers, which are usually used in EEPROM emulation, can be used to distinguish expected valid data from unknown data fields. Only blank failures in Complement Sensing mode are reported in this flag and is intended for use only in this mode.</p> <p><i>Note:</i> Reset by hardware when <b>SF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> Read data cells are not erased.  <math>1_B</math> Read data cells have sufficient high current: a program of new data without prior erase is possible.</p>

Field	Bits	Type	Description
<b>AER1</b>	16	rh	<p><b>Accumulated Single Bit ECC Errors</b>            The status flag accumulates single bit failures during NVM read operations.</p> <p><i>Note:</i> Reset by hardware when <b>SF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> No single bit ECC failure occurred.  <math>1_B</math> At least one single bit ECC failure occurred.</p>
<b>AER2</b>	17	rh	<p><b>Accumulated Double Bit ECC Errors</b>            The status flag accumulates double bit failures during NVM read operations.</p> <p><i>Note:</i> Reset by hardware when <b>SF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> No double bit ECC failure occurred.  <math>1_B</math> At least one double bit ECC failure occurred.</p>
<b>AER3</b>	18	rh	<p><b>Accumulated Triple Bit ECC Errors</b>            The status flag accumulates triple bit failures during NVM read operations.</p> <p><i>Note:</i> Reset by hardware when <b>SF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> No triple bit ECC failure occurred.  <math>1_B</math> At least one triple bit ECC failure occurred.</p>
<b>AERM</b>	19	rh	<p><b>Accumulated Multi-bit ECC Errors</b>            The status bit accumulates multi bit failures during NVM read accesses.</p> <p><i>Note:</i> Reset by hardware when <b>SF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> No multi bit ECC failure occurred.  <math>1_B</math> Multi bit ECC failure occurred.</p>
<b>AERANY</b>	23	rh	<p><b>Accumulated Any Read Access ECC Error</b>            The status bit accumulates ECC failures during NVM read accesses.</p> <p><i>Note:</i> Reset by hardware when <b>SF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> No ECC failure occurred.  <math>1_B</math> ECC failure occurred.</p>
<b>R24</b>	24	rX	<b>Reserved - RES</b>

Field	Bits	Type	Description
<b>ABLANKA</b>	25	rh	<p><b>Accumulated Blank Analog</b></p> <p>The flag accumulates analog evaluated blank failures during NVM read accesses. It reports that all read data cells have sufficient high current: a program of new data without prior erase is possible. Under certain operation history, a valid complement data entry may also appear as blank. Data qualifiers like headers or footers, which are usually used in EEPROM emulation, can be used to distinguish expected valid data from unknown data fields. Only blank failures in Complement Sensing mode are reported in this flag and is intended for use only in this mode.</p> <p><i>Note:</i> Reset by hardware when <b>SF_ECCC.CLR</b> is written to <math>11_B</math>.</p> <p><math>0_B</math> Read data cells are not erased.  <math>1_B</math> Read data cells have sufficient high current: a program of new data without prior erase is possible.</p>

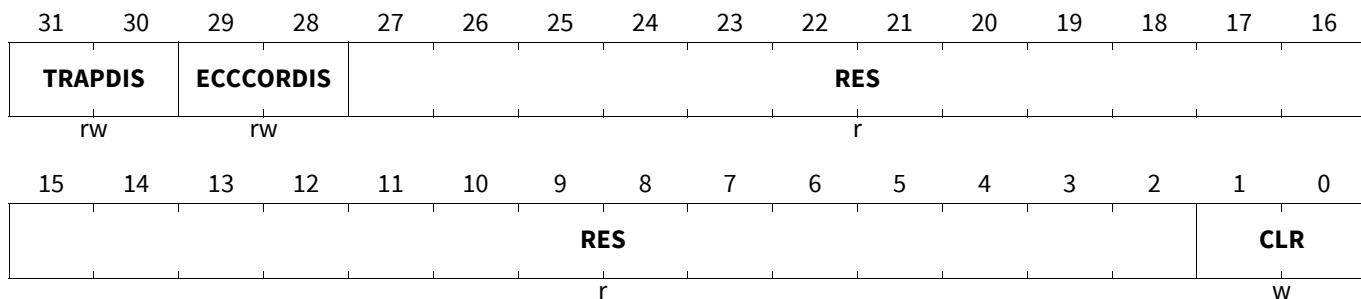
### HSM DF1 ECC Control Register

#### SF\_ECCC

##### HSM DF1 ECC Control Register

(0020048<sub>H</sub>)

Application Reset Value: C000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>CLR</b>	1:0	w	<p><b>Clear ECC status bits</b></p> <p><math>00_B</math> No action.  ...  <math>10_B</math> No action.  <math>11_B</math> Clear the DMU_SF_ECCS status bits.</p>
<b>RES</b>	27:2	r	<p><b>Reserved</b></p> <p>Always read as 0; should be written with 0.</p>
<b>ECCCORDIS</b>	29:28	rw	<p><b>HSM Command Interface ECC Correction Disable</b></p> <p><math>00_B</math> <b>Enabled</b>, If DF1 is configured as HSM_exclusive, then, ECC correction for the DF1 read path is enabled.  ...  <math>10_B</math> <b>Enabled</b>, If DF1 is configured as HSM_exclusive, then, ECC correction for the DF1 read path is enabled.  <math>11_B</math> <b>Disabled</b>, If DF1 is configured as HSM_exclusive, then, ECC correction for the DF1 read path is disabled.</p>

Field	Bits	Type	Description
<b>TRAPDIS</b>	31:30	rw	<p><b>HSM Command Interface Uncorrectable ECC Bit Error Trap Disable</b></p> <p>00<sub>B</sub> If DF1 is configured as HSM_exclusive, and if an uncorrectable ECC error occurs, then a bus error trap is generated.</p> <p>...</p> <p>10<sub>B</sub> If DF1 is configured as HSM_exclusive, and if an uncorrectable ECC error occurs, then a bus error trap is generated.</p> <p>11<sub>B</sub> If DF1 is configured as HSM_exclusive, then the uncorrectable ECC error trap is disabled.</p>

# HSM DF1 ECC Write Register

The HSM ECC Write Register contains bits for disabling the ECC encoding separately for DF1. When disabling the ECC encoding with **SF\_ECCW.ECCENCDIS** = '11<sub>B</sub>' the ECC code for the next 256-bit data block transferred from DMU to the Flash assembly buffer is taken from **SF\_ECCW.WCODE**. When **SF\_ECCW.ECCENCDIS** is set to '11<sub>B</sub>' the "Write Burst" command sequence must not be used as it may result in unpredictable results.

**Attention:** After reading data with disabled ECC correction data buffers it is recommended to perform a reset to resume normal operation with ECC correction

Field	Bits	Type	Description
<b>WCODE</b>	21:0	rw	<b>Error Correction Write Code</b> 22-bit ECC code for the current 64-bit (for DFLASH) or 256-bit (for PFLASH) write buffer to be written into the assembly buffer instead of a generated ECC.
<b>RES</b>	29:22	r	<b>Reserved</b> Always read as 0.
<b>ECCENCDIS</b>	31:30	rw	<b>ECC Encoding Disable</b>  00 <sub>B</sub> The ECC code is automatically calculated. ... 10 <sub>B</sub> The ECC code is automatically calculated. 11 <sub>B</sub> The ECC code is taken from WCODE.

#### **6.5.3.2.17 Data Flash Bank 1 Mode Control Registers**

## HSM DF1 User Mode Control

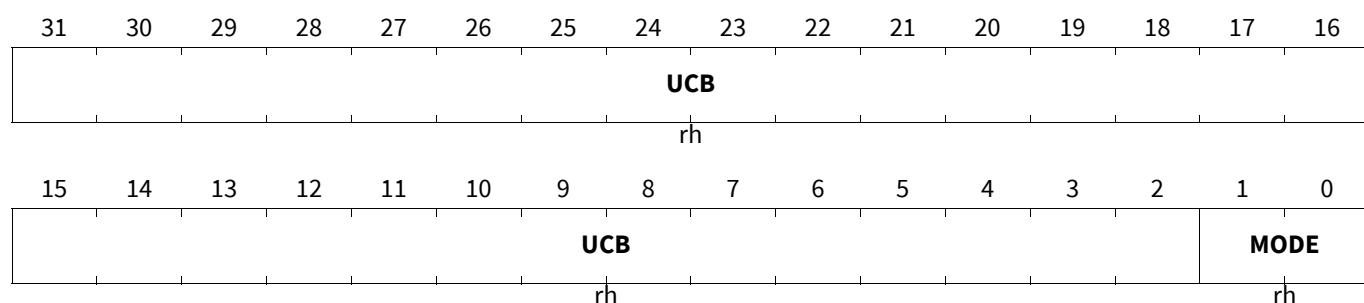
The DF1 Protection Configuration User Mode Control Register is loaded from UCB during startup.

SF PROCONUSR

HSM DF1 User Mode Control

(0020074\_u)

#### **Reset Value: Table 187**



Field	Bits	Type	Description
<b>MODE</b>	1:0	rh	<p><b>DF1 User Mode Control</b></p> <p>Configures the DF1 mode when the user has control.</p> <ul style="list-style-type: none"> <li><math>00_B</math> Single Ended.</li> <li><math>01_B</math> Complement Sensing.</li> <li>...</li> <li><math>11_B</math> Complement Sensing.</li> </ul>
<b>UCB</b>	31:2	rh	<p><b>Reserved for UCB</b></p> <p>Deliver the corresponding content of UCB.</p>

**Table 187 Reset Values of SF\_PROCONUSR**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### **6.5.3.2.18 HSM Suspend**

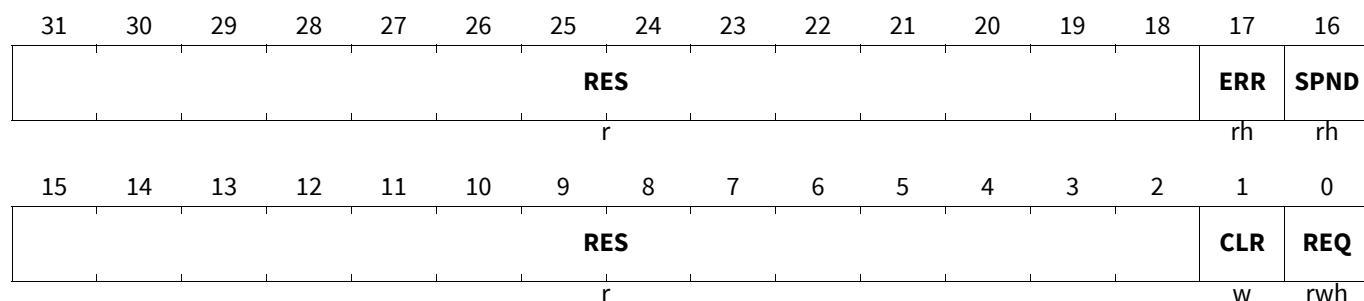
## HSM Suspend Control Register

SF SUSPEND

## **HSM Suspend Control Register**

(00200E8<sub>H</sub>)

**Application Reset Value: 0000 0000**



Field	Bits	Type	Description
<b>REQ</b>	0	rwh	<b>Suspend Request</b> 0 <sub>B</sub> No action. 1 <sub>B</sub> Suspension of a Flash operation requested or pending.
<b>CLR</b>	1	w	<b>Suspend Clear</b> Software write only active high clear of Suspend Error. 0 <sub>B</sub> No action. 1 <sub>B</sub> Clear Suspend Error status.
<b>RES</b>	15:2, 31:18	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>SPND</b>	16	rh	<b>Flash Operation Suspended</b> Suspension of a Flash program or erase operation. 0 <sub>B</sub> No Flash operation is suspended. 1 <sub>B</sub> Flash operation is suspended.
<b>ERR</b>	17	rh	<b>Suspend Error</b> <i>Note:</i> Reset by hardware when <b>SF_SUSPEND.CLR</b> is written to 1 <sub>B</sub> . 0 <sub>B</sub> No suspend error. 1 <sub>B</sub> Last suspend request via DMU_SF_SUSPEND failed.

### 6.5.3.2.19 Margin Check Control

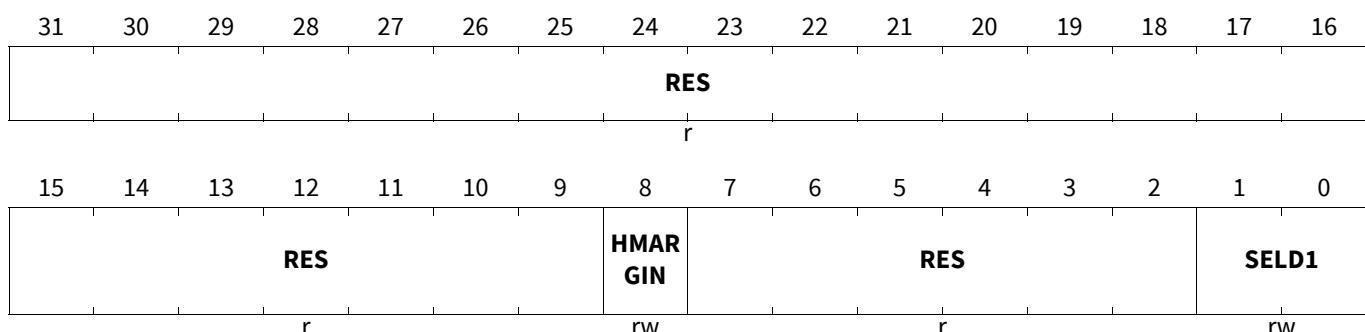
#### HSM DF1 Margin Control Register

##### SF\_MARGIN

##### HSM DF1 Margin Control Register

(00200EC<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>SELD1</b>	1:0	rw	<p><b>DF1 Margin Read Selection</b></p> <p><b>Note:</b> If a change between the standard and hard read margin setting is done, the system must wait a delay time <math>tFL\_MarginDel</math> until the next read is requested.</p> <p><math>00_B</math> Read with standard margin.  <math>\dots</math>  <math>10_B</math> Read with standard margin.  <math>11_B</math> Read with hard margin selected by DMU_SF_MARGIN.HMARGIN</p>
<b>RES</b>	7:2, 31:9	r	<p><b>Reserved</b>  Always read as 0; should be written with 0.</p>
<b>HMARGIN</b>	8	rw	<p><b>Hard Margin Selection</b></p> <p><b>Note:</b> Suboptimal 0-bits are read as 1s.</p> <p><b>Note:</b> Suboptimal 1-bits are read as 0s.</p> <p>The concrete margin values are restored from the configuration sector and are determined by Infineon.</p> <p><math>0_B</math> <b>Tight0</b>, Tight margin for 0 (low) level.  <math>1_B</math> <b>Tight1</b>, Tight margin for 1 (high) level.</p>

### 6.5.3.2.20 HSM OTP Protection Configuration

The configuration of HSM Code read/write/OTP protection is indicated with the HSM OTP registers.

#### HSM Protection Configuration

**SP\_PROCONHSMCFG** represents after Flash startup the or-combination of all boot sector selection entries stored in the HSMCOTP configuration sets.

#### SP\_PROCONHSMCFG

##### HSM Protection Configuration

(0030000<sub>H</sub>)

Reset Value: [Table 188](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB	BLKFL AN	DESTDBG		HSMENRES	HSMENPINS	UCB	HSMRAMKEE P	UCB	HSMD X	SSWW AIT	HSMB OOTE N				
rh	rh	rh		rh	rh	rh	rh	rh	rh	rh	rh				rh

Field	Bits	Type	Description
<b>HSMBOOTEN</b>	0	rh	<b>HSM Boot Enable</b> $0_B$ HSM Boot is not enabled. $1_B$ HSM Boot is enabled.
<b>SSWWAIT</b>	1	rh	<b>SSW Wait</b> Defines if the SSW waits for the HSM to release the jump of CPU0 to user code. $0_B$ The SSW does not wait for HSM. $1_B$ SSW wait for acknowledge of HSM.
<b>HSMDX</b>	2	rh	<b>HSM Data Sectors Exclusive</b> This bit indicates whether the DFLASH1 logical sectors EEPROMx are configured as “HSM_exclusive”. $0_B$ HSMx are not HSM_exclusive. $1_B$ HSMx are HSM_exclusive.
<b>UCB</b>	3, 6, 15:14	rh	<b>Reserved for UCB</b> Deliver the corresponding content of UCB_HSMCOTP.
<b>HSMRAMKEEP</b>	5:4	rh	<b>HSM RAM Clear</b> $00_B$ SSW clears HSM RAM after all power-on-resets and system resets. $01_B$ SSW clears HSM RAM after all power-on-resets but not system reset. $10_B$ SSW clears HSM RAM after all power-on-resets but not system reset. $11_B$ SSW clears HSM RAM only after cold power-on-reset.
<b>HSMENPINS</b>	8:7	rh	<b>Enable HSM Forcing of Pins HSM1/2</b> This bit indicates whether HSM may force the value of the pins HSM1/2 (i.e. overrule the value driven by the application). $00_B$ HSM cannot force pins. ... $10_B$ HSM cannot force pins. $11_B$ HSM can force pins.
<b>HSMENRES</b>	10:9	rh	<b>Enable HSM Triggering Resets</b> This bit indicates whether HSM may trigger application or system resets. $00_B$ HSM cannot trigger resets. ... $10_B$ HSM cannot trigger resets. $11_B$ HSM can trigger resets.
<b>DESTDBG</b>	12:11	rh	<b>Destructive Debug Entry</b> This field configures the destructive debug entry. $00_B$ Debug entry is non-destructive. ... $10_B$ Debug entry is non-destructive. $11_B$ Debug entry is destructive.

Field	Bits	Type	Description
<b>BLKFLAN</b>	13	rh	<b>Block Flash Analysis</b> The commands “Verify Erased Page”, “Verify Erased WL”, “Verify Erased Logical Sector Range”; are blocked on the HSM code ranges. $0_B$ Functions allowed on all Flash ranges. $1_B$ Functions blocked on HSM_exclusive Flash ranges.
<b>RES</b>	31:16	r	<b>Reserved</b> Always read as 0; should be written with 0.

**Table 188 Reset Values of SP\_PROCONHSMCFG**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**HSM Code Boot Sector**

**SP\_PROCONHSMCBS** represents after Flash startup the or-combination of all boot sector selection entries stored in the HSMCOTP configuration sets.

**SP\_PROCONHSMCBS****HSM Code Boot Sector**(0030004<sub>H</sub>)**Reset Value: Table 189**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>UCB</b>								<b>UCB</b>							
r				rh				r				rh			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>UCB</b>								<b>UCB</b>							
rh				rh				rh				rh			

Field	Bits	Type	Description
<b>BOOTSEL0</b>	5:0	rh	<b>Boot Sector Selection</b> This field controls which of the HSM code sectors is searched for boot code. See <a href="#">Table 190</a> .
<b>UCB</b>	7:6, 15:14	rh	<b>Reserved for UCB</b> Deliver the corresponding content of UCB_HSMCOTP.
<b>BOOTSEL1</b>	13:8	rh	<b>Boot Sector Selection</b> This field controls which of the HSM code sectors is searched for boot code. See <a href="#">Table 190</a> .
<b>BOOTSEL2</b>	21:16	rh	<b>Boot Sector Selection</b> This field controls which of the HSM code sectors is searched for boot code. See <a href="#">Table 190</a> .

Field	Bits	Type	Description
<b>UCB</b>	23:22, 31:30	r	<b>Reserved for UCB</b> Deliver the corresponding content of UCB_HSMCOTP.
<b>BOOTSEL3</b>	29:24	rh	<b>Boot Sector Selection</b> This field controls which of the HSM code sectors is searched for boot code. See <a href="#">Table 190</a> .

**Table 189 Reset Values of SP\_PROCONHSMCBS**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**Boot Sector Selection**

The Boot Sector Selection controls which of the HSM code sectors is searched for boot code.

**Table 190 Boot Sector Selection**

BOOTSELx (x = 0 - 3)	Description
00 <sub>H</sub>	Sector HSM0X is searched.
01 <sub>H</sub>	Sector HSM1X is searched.
02 <sub>H</sub>	Sector HSM2X is searched.
03 <sub>H</sub>	Sector HSM3X is searched.
04 <sub>H</sub>	Sector HSM4X is searched.
05 <sub>H</sub>	Sector HSM5X is searched.
06 <sub>H</sub>	Sector HSM6X is searched.
07 <sub>H</sub>	Sector HSM7X is searched.
08 <sub>H</sub>	Sector HSM8X is searched.
09 <sub>H</sub>	Sector HSM9X is searched.
0A <sub>H</sub>	Sector HSM10X is searched.
0B <sub>H</sub>	Sector HSM11X is searched.
0C <sub>H</sub>	Sector HSM12X is searched.
0D <sub>H</sub>	Sector HSM13X is searched.
0E <sub>H</sub>	Sector HSM14X is searched.
0F <sub>H</sub>	Sector HSM15X is searched.
10 <sub>H</sub>	Sector HSM16X is searched.
11 <sub>H</sub>	Sector HSM17X is searched.
12 <sub>H</sub>	Sector HSM18X is searched.
13 <sub>H</sub>	Sector HSM19X is searched.
14 <sub>H</sub>	Sector HSM20X is searched.
15 <sub>H</sub>	Sector HSM21X is searched.
16 <sub>H</sub>	Sector HSM22X is searched.

**Table 190 Boot Sector Selection (cont'd)**

<b>BOOTSELx (x = 0 - 3)</b>	<b>Description</b>
17 <sub>H</sub>	Sector HSM23X is searched.
18 <sub>H</sub>	Sector HSM24X is searched.
19 <sub>H</sub>	Sector HSM25X is searched.
1A <sub>H</sub>	Sector HSM26X is searched.
1B <sub>H</sub>	Sector HSM27X is searched.
1C <sub>H</sub>	Sector HSM28X is searched.
1D <sub>H</sub>	Sector HSM29X is searched.
1E <sub>H</sub>	Sector HSM30X is searched.
1F <sub>H</sub>	Sector HSM31X is searched.
20 <sub>H</sub>	Sector HSM32X is searched.
21 <sub>H</sub>	Sector HSM33X is searched.
22 <sub>H</sub>	Sector HSM34X is searched.
23 <sub>H</sub>	Sector HSM35X is searched.
24 <sub>H</sub>	Sector HSM36X is searched.
25 <sub>H</sub>	Sector HSM37X is searched.
26 <sub>H</sub>	Sector HSM38X is searched.
27 <sub>H</sub>	Sector HSM39X is searched.
Others	Reserved.

### HSM Code Exclusive Protection Configuration

The register indicates if a PFLASH logical sector is configured HSM\_exclusive. This register represents after Flash startup the or-combination of all PROCONHSMX entries stored in the HSMCOTP configuration sets.

#### SP\_PROCONHSMCX0

#### HSM Code Exclusive Protection Configuration (0030008<sub>H</sub>)

Reset Value: [Table 191](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>HSM3 1X</b>	<b>HSM3 0X</b>	<b>HSM2 9X</b>	<b>HSM2 8X</b>	<b>HSM2 7X</b>	<b>HSM2 6X</b>	<b>HSM2 5X</b>	<b>HSM2 4X</b>	<b>HSM2 3X</b>	<b>HSM2 2X</b>	<b>HSM2 1X</b>	<b>HSM2 0X</b>	<b>HSM1 9X</b>	<b>HSM1 8X</b>	<b>HSM1 7X</b>	<b>HSM1 6X</b>
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>HSM1 5X</b>	<b>HSM1 4X</b>	<b>HSM1 3X</b>	<b>HSM1 2X</b>	<b>HSM1 1X</b>	<b>HSM1 0X</b>	<b>HSM9 X</b>	<b>HSM8 X</b>	<b>HSM7 X</b>	<b>HSM6 X</b>	<b>HSM5 X</b>	<b>HSM4 X</b>	<b>HSM3 X</b>	<b>HSM2 X</b>	<b>HSM1 X</b>	<b>HSM0 X</b>
rh															

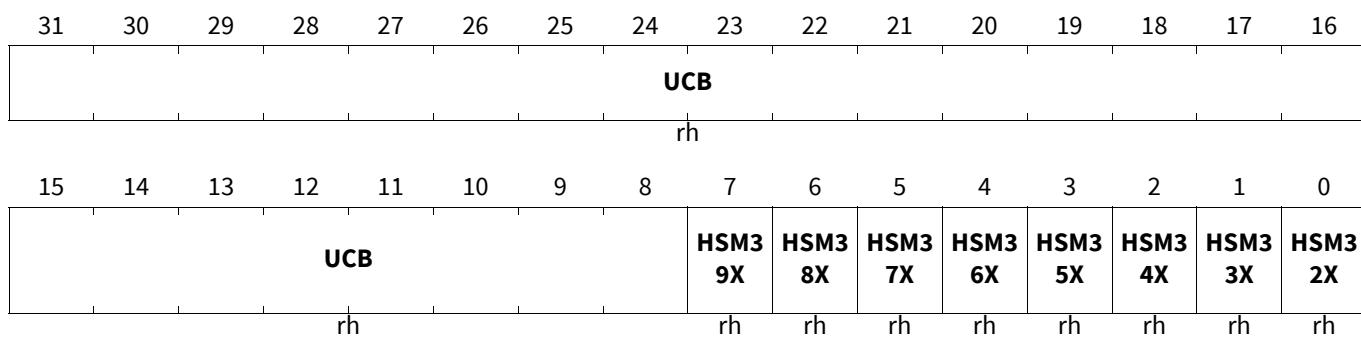
<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>HSMxX (x=0-31)</b>	x	rh	<b>PFLASH Sector x HSM Code Exclusive</b> This bit indicates whether the PFLASH logical sector is "HSM_exclusive". 0 <sub>B</sub> Logical sector is not configured HSM_exclusive. 1 <sub>B</sub> Logical sector is configured HSM_exclusive.

**Table 191 Reset Values of SP\_PROCONHSMCX0**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**HSM Code Exclusive Protection Configuration**

The register indicates if a PFLASH logical sector is configured HSM\_exclusive. This register represents after Flash startup the or-combination of all PROCONHSMX entries stored in the HSMCOTP configuration sets.

**SP\_PROCONHSMCX1****HSM Code Exclusive Protection Configuration (003000C<sub>H</sub>)****Reset Value: Table 192**

Field	Bits	Type	Description
HSMxX (x=32-39)	x-32	rh	<b>PFLASH Sector x HSM Code Exclusive</b> This bit indicates whether the PFLASH logical sector is “HSM_exclusive”. 0 <sub>B</sub> Logical sector is not configured HSM_exclusive. 1 <sub>B</sub> Logical sector is configured HSM_exclusive.
UCB	31:8	rh	<b>Reserved for UCB</b> Deliver the corresponding content of UCB_HSMCOTP.

**Table 192 Reset Values of SP\_PROCONHSMCX1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**HSM Code OTP Protection Configuration**

The register indicates if a PFLASH logical sector is configured HSM locked forever. This register represents after Flash startup the or-combination of all PROCONHSMOTP entries stored in the HSMCOTP configuration sets.

**SP\_PROCONHSMCOTP0****HSM Code OTP Protection Configuration (0030010<sub>H</sub>)**Reset Value: [Table 193](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HSM3 1ROM	HSM3 0ROM	HSM2 9ROM	HSM2 8ROM	HSM2 7ROM	HSM2 6ROM	HSM2 5ROM	HSM2 4ROM	HSM2 3ROM	HSM2 2ROM	HSM2 1ROM	HSM2 0ROM	HSM1 9ROM	HSM1 8ROM	HSM1 7ROM	HSM1 6ROM
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSM1 5ROM	HSM1 4ROM	HSM1 3ROM	HSM1 2ROM	HSM1 1ROM	HSM1 0ROM	HSM9 ROM	HSM8 ROM	HSM7 ROM	HSM6 ROM	HSM5 ROM	HSM4 ROM	HSM3 ROM	HSM2 ROM	HSM1 ROM	HSM0 ROM
rh															

Field	Bits	Type	Description
HSMxROM (x=0-31)	x	rh	<b>PFLASH Sector x HSM Code Locked Forever</b> This bit indicates whether PFLASH sector is an HSM Code OTP protected sector with read-only functionality. 0 <sub>B</sub> No OTP protection is configured. 1 <sub>B</sub> OTP protection is configured.

**Table 193 Reset Values of SP\_PROCONHSMCOTP0**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

**HSM Code OTP Protection Configuration**

The register indicates if a PFLASH logical sector is configured HSM locked forever. This register represents after Flash startup the or-combination of all PROCONHSMOTP entries stored in the HSMCOTP configuration sets.

**SP\_PROCONHSMCOTP1****HSM Code OTP Protection Configuration (0030014<sub>H</sub>)**Reset Value: [Table 194](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB															
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB								HSM3 9ROM	HSM3 8ROM	HSM3 7ROM	HSM3 6ROM	HSM3 5ROM	HSM3 4ROM	HSM3 3ROM	HSM3 2ROM
rh								rh							

Field	Bits	Type	Description
HSMxROM (x=32-39)	x-32	rh	<b>PFLASH Sector x HSM Code Locked Forever</b> This bit indicates whether PFLASH sector is an HSM Code OTP protected sector with read-only functionality. 0 <sub>B</sub> No OTP protection is configured. 1 <sub>B</sub> OTP protection is configured.
UCB	31:8	rh	<b>Reserved for UCB</b> Deliver the corresponding content of UCB_HSMCOTP.

Table 194 Reset Values of SP\_PROCONHSMCOTP1

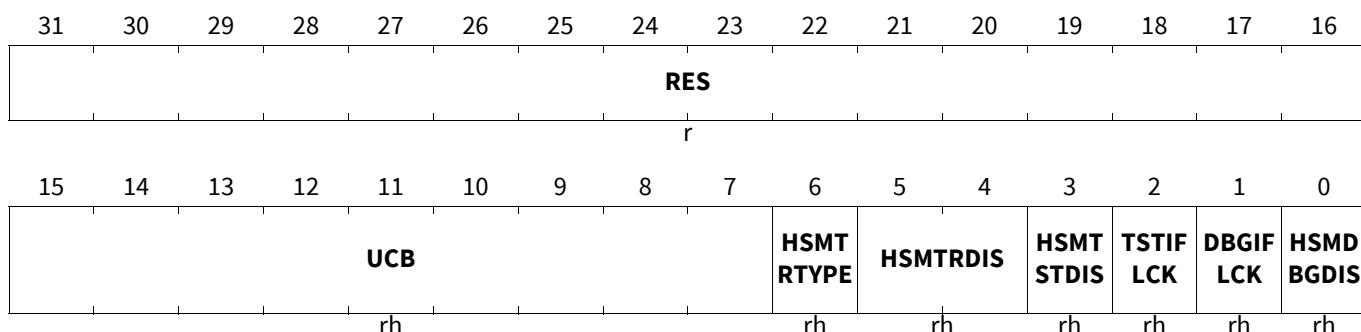
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### 6.5.3.2.21 HSM Interface Protection Configuration

A disabled HSM debug access can be reopened temporarily or permanently by the HSM only after successful authorisation

#### HSM Interface Protection Configuration

**SP\_PROCONHSM**  
**HSM Interface Protection Configuration (0030040<sub>H</sub>)** Reset Value: [Table 195](#)



Field	Bits	Type	Description
HSMDBGDIS	0	rh	<b>HSM Debug Disable</b> This bit indicates whether HSM debug is configured as “disabled”. 0 <sub>B</sub> HSM debug is enabled. 1 <sub>B</sub> HSM debug is disabled.
DBGIFLCK	1	rh	<b>Debug Interface Locked</b> This bit indicates whether the chip debug interface is configured as “locked”. 0 <sub>B</sub> Debug is unlocked. 1 <sub>B</sub> Debug is locked.

Field	Bits	Type	Description
TSTIFLCK	2	rh	<p><b>Test Interface Locked</b></p> <p>This bit indicates whether the chip test interface is configured as “locked”.</p> <p><math>0_B</math> Test interface is unlocked. <math>1_B</math> Test interface is locked.</p>
HSMTSTDIS	3	rh	<p><b>HSM Test Disable</b></p> <p>This bit indicates whether the HSM test is configured as “disabled”.</p> <p><math>0_B</math> HSM test is enabled. <math>1_B</math> HSM test is disabled.</p>
HSMTRDIS	5:4	rh	<p><b>HSM Trace Disable</b></p> <p>This bit field indicates whether the HSM tracing and capturing of transactions for debug and in error cases via BCU is configured as “disabled”.</p> <p><i>Note:</i> <i>In order to ensure that with UCB delivery state HSM tracing is fully enabled the encoding for this bit field is inverted compared to the corresponding HSM control registers.</i></p> <p><math>00_B</math> <b>TRACEEN</b>, Tracing enabled. <math>01_B</math> <b>TRACEDIS2</b>, Tracing disabled. ... <math>11_B</math> <b>TRACEDIS0</b>, Tracing disabled.</p>
HSMTRTYPE	6	rh	<p><b>HSM Type of Trace</b></p> <p>This bit field indicates which information can be captured by the BCU for HSM transactions.</p> <p><i>Note:</i> <i>In order to ensure that with UCB delivery state HSM tracing is fully enabled the encoding for this bit field is inverted compared to the corresponding HSM control registers.</i></p> <p><math>0_B</math> <b>TRDATA</b>, Trace addresses and data. <math>1_B</math> <b>TRADDR</b>, Trace addresses only.</p>
UCB	15:7	rh	<p><b>Reserved for UCB</b></p> <p>Deliver the corresponding content of UCB_HSM.</p>
RES	31:16	r	<p><b>Reserved</b></p> <p>Always read as 0; should be written with 0.</p>

Table 195 Reset Values of SP\_PROCONHSM

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

## 6.5.4 Security

The security protection ensures that the PFLASH banks cannot be disturbed by software crashes or accesses by “unsafe” masters. This is achieved by the following features:

- Safety ENDINIT protection of ACCEN registers.
- Master specific protection of Flash relevant control and configuration registers.

The security protection prevents unauthorized PFLASH read and write accesses from internal masters or from external masters:

- Flash read protection: protected Flash sections can only be read when booting from Flash.
- Flash write protection: protected Flash sections can only be changed after authentication.
- Flash OTP (One-Time Programmable) protection: these Flash sections can't be changed anymore.
- HSM specific protection: it ensures that the HSM can protect its private data from access by other software. Additionally selected data can only be read during HSM boot phase.
- HSM debug protection: when the HSM allows internal debugging, accesses by the debug master (Cerberus) have the same privileges as the HSM itself.
- Device debug protection: depending on the boot mode and the configuration of debug protection and the Flash read protection and further controlled by HSM the debug access to the device is enabled.

The following security protection guards the Flash against mal-operation including software crashes (even of “safe” masters):

- ENDINIT and SV mode protection of configuration registers.
- Command sequences that can change Flash content need more command cycles. DMU\_HF\_ERRSR.SQER stops command interpretation.
- The separate HSM command interface enables main CPUs and HSM to share the DFLASH without disturbing each other and without the need for coordinating the software drivers.

The security protection supplements the Functional Safety Features because Flash data protected by the write protection is also safe against software crashes.

The following table gives an overview of all protection features and for which resources they are effective.

**Table 196 Security Protection**

	PFLASH	DFLASH
Flash Write Protection	Restricts Command Sequences	Restricts Command Sequences
Flash Read Protection	Restricts Reads	Restricts Reads

Any access to these resources has to pass all checks before the operation is performed.

### 6.5.4.1 Effective Flash Read Protection

The Flash read protection depends on the master and the Flash address range.

The following Flash ranges have separate read protections:

- PFLASH:
  - PFLASH Banks PFp including Erase Counters ECp
  - HSM Code exclusive sectors
- DFLASH:
  - CPU EEPROM
  - UCBs

- CFS
- HSM EEPROM

#### **6.5.4.1.1 PFLASH Read Protection**

All read accesses to a PFLASH bank are routed via the local CPU. Master specific access control (including local CPU PMBI and DMBI accesses) is configured and controlled by the CPUx ACCEN1 and ACCEN0 registers.

A read access to PFLASH (with the exception of HSM\_exclusive sectors) fails with bus error under the following conditions:

- DMU\_HF\_CONTROL.DDFP is 1<sub>B</sub>

Activating Read Protection:

- DMU\_HF\_CONTROL.DDFP is initialized by SSW depending on the startup mode and configured protection.
- DMU\_HF\_CONTROL.DDFP can be directly modified by user software under conditions noted in the description of DMU\_HF\_CONTROL

#### **HSM Code Read Protection**

The read access to the PFLASH Bank logical sector configured as an HSM code sectors depends on the setting of DMU\_SP\_PROCONHSMCX0.HSMxX or DMU\_SP\_PROCONHSMCX1.HSMxX which defines the “HSM\_exclusive” attribute of these sectors.

If DMU\_SP\_PROCONHSMCX0.HSMxX or DMU\_SP\_PROCONHSMCX1.HSMxX is set for such a sector then:

- HSM: full read access.
- Cerberus: with enabled HSM debug same access rights as HSM itself, with disabled HSM debug access rights as all other masters.
- All other masters except HSM and Cerberus with HSM debug access rights: a read fails with a bus error.

If DMU\_SP\_PROCONHSMCX0.HSMxX or DMU\_SP\_PROCONHSMCX1.HSMxX is cleared for such a sector then:

- The same rules as for the other PFLASH sectors based on CPU access enable as described above are valid.

#### **6.5.4.1.2 DFLASH Read Protection**

A read access to DFLASH0\_EEPROM fails with bus error under the following conditions:

- DMU\_HF\_CONTROL.DDFD is 1<sub>B</sub>

Activating Read Protection:

- DMU\_HF\_CONTROL.DDFD is initialized by SSW depending on the startup mode and configured protection.
- DMU\_HF\_CONTROL.DDFD can be directly modified by user software under conditions noted in the description of DMU\_HF\_CONTROL

#### **HSM EEPROM Read Protection**

The read access to the HSM data sectors HSMx depends on the configuration of DMU\_SP\_PROCONHSMCFG.HSMDX.

If DMU\_SP\_PROCONHSMCFG.HSMDX is set:

- HSM: full read access.
- Cerberus: with enabled HSM debug same access rights as HSM itself, with disabled HSM debug access rights as all other masters.
- All other masters except HSM and Cerberus with HSM debug access rights: a read fails with a bus error.

If DMU\_SP\_PROCONHSMCFG.HSMDX is cleared read access is allowed for all masters.

### 6.5.4.2 Effective Flash Write Protection

A range of Flash can be write protected by several means that are all configured in the UCBs.

#### 6.5.4.2.1 PFLASH Write Protection

Programming, erasing and replacing (using Replace Logical Sector command sequence) of a group of PFLASH logical sectors (with the exception of “HSM\_exclusive” code sectors) fails with PROER if any of the following conditions is true:

- DMU\_HF\_PROCONPF.RPRO and not (DMU\_HF\_PROTECT.PRODISP (global write protection disable) or DMU\_HF\_PROTECT.PRODISP0-5 (sector specific write protection disable)).
- DMU\_HP\_PROCONPp.SxL and not (DMU\_HF\_PROTECT.PRODISP or DMU\_HF\_PROTECT.PRODISP0-5).
- DMU\_HP\_PROCONOTPp.SxROM (sector specific OTP protection).
  - A set DMU\_HP\_PROCONOTPp.SxROM prohibits write and erase commands. These fail with DMU\_HF\_ERRSR.PROER.
- DMU\_HP\_PROCONWOPp.SxWOP (sector specific write-once protection).
  - A set DMU\_HP\_PROCONWOPp.SxWOP prohibits “Write Page”, “Write Burst”, “Replace Logical Sector” and erase commands. These fail with DMU\_HF\_ERRSR.PROER.
  - Only “Write Page Once” or “Write Burst Once” is accepted by the DMU. The FSI checks if the addressed Flash range is erased. If this is not the case the FSI reports a PVER and doesn’t start programming.
- Writes to PFLASH are Safety Endinit protected.

#### HSM Code Write Protection

Programming, erasing and replacing (using Replace Logical Sector command sequence) of the HSM exclusive logical sectors depends on the setting of the “HSM\_exclusive” attribute of these sectors and which master performs the access.

The above mentioned operations on “HSM\_exclusive” sectors fails if any of the following conditions is true:

- For all masters except HSM: DMU\_SP\_PROCONHSMCX0.HSMxX or DMU\_SP\_PROCONHSMCX1.HSMxX
- HSM specific OTP protection: DMU\_SP\_PROCONHSMCOTP0.HSMxROM or DMU\_SP\_PROCONHSMCOTP1.HSMxROM
- Writes to PFLASH are Safety Endinit protected.

For Cerberus the following rules apply: with enabled HSM debug same access rights as HSM itself, with disabled HSM debug access rights as all other masters.

#### 6.5.4.2.2 DFLASH Write Protection

Programming and erasing of the DFLASH0\_EEPROM fails with PROER when any of the following conditions is true:

- DMU\_HF\_PROCONDF.RPRO and not DMU\_HF\_PROTECT.PRODISD.
- DMU\_HF\_PROCONDF.L and not DMU\_HF\_PROTECT.PRODISD.

#### HSM EEPROM Write Protection

Programming and erasing of the HSM data sectors depends on DMU\_SP\_PROCONHSMCFG.HSMDX which defines their “HSM\_exclusive” attribute.

If DMU\_SP\_PROCONHSMCFG.HSMDX is set:

- HSM: full write access.
- All masters except HSM: programming and erasing fails.

- Cerberus: with enabled HSM debug same access rights as HSM itself, with disabled HSM debug access rights as all other masters.

If DMU\_SP\_PROCONHSMCFG.HSMDX is cleared:

- All master can program and erase without restriction.

### 6.5.4.3 Configuring Protection in the UCB

The effective protection is determined by the content of the PROCONx registers. These are loaded during startup from the UCBs. A UCB is erased with the command “Erase Logical Sector Range”. Its pages can be programmed with “Write Page” or “Write Burst”. Each UCB has its own access control. When programming or erasing fail because of this access control PROER is set. When reading fails a bus error is returned.

#### 6.5.4.3.1 UCB Confirmation

The state of a UCB is determined by the confirmation code:

**Table 197 UCB States**

State	Value	Description
UNLOCKED	4321 1234 <sub>H</sub>	<b>Delivery State</b> The UCB confirmation code is programmed with the UNLOCKED value.
CONFIRMED	57B5 327F <sub>H</sub>	<b>Operational State</b> The UCB confirmation code is programmed with the CONFIRMED value.  <i>Note: The UNLOCKED value can be over programmed with the CONFIRMED value.</i>
ERASED	0000 0000 <sub>H</sub>	<b>Erased State</b> Behavior as for the ERRORED state.
ERRORED	Others	<b>Errorred State</b> The UCB confirmation code stored is not the CONFIRMED or UNLOCKED value.

As also the erased state is considered as **ERRORED** the transition from **UNLOCKED** to **CONFIRMED** state can be done without erasing the UCB. For this the **UNLOCKED** code in the pages with the confirmation code and the copied confirmation code can be over-programmed with 57B5 327F<sub>H</sub>. It has to be ensured that the 4 bytes following the confirmation code (e.g. at offset 1F4<sub>H</sub>) are kept 0000 0000<sub>H</sub> in the unlocked state and in the over-programmed data. Only then the result after over-programming is ECC clean.

#### Dual UCB: Confirmation States

The UCB content is split across separate ORIG and COPY UCBs (e.g. UCB\_PFLASH\_ORIG and UCB\_PFLASH\_COPY). The UCB confirmation state is derived from the ORIG and COPY UCB confirmation codes.

The UCB confirmation is **UNLOCKED** if one of the following confirmation state conditions is true:

- ORIG UCB confirmation code is **UNLOCKED**.
- ORIG UCB confirmation code is **ERRORED** and the COPY UCB confirmation code is **UNLOCKED**.

The UCB confirmation is **CONFIRMED** if one of the following confirmation state conditions is true:

- ORIG UCB confirmation code is **CONFIRMED**.
- ORIG UCB confirmation code is **ERRORED** and the COPY UCB confirmation code is **CONFIRMED**.

The UCB confirmation is **ERRORED** if one of the following confirmation state conditions is true:

- ORIG UCB confirmation code is **ERRORED** and the COPY UCB confirmation code is **ERRORED**.

### **Single UCB: Errored State or ECC Error in the UCB Confirmation Codes**

If a UCB contains both ORIG and COPY confirmation codes (e.g. UCB\_SSW) then the following status is reported.

If the ORIG confirmation code is an **ERRORED** value or contains an uncorrectable ECC error then:

- The COPY confirmation code will be read to determine the UCB confirmation state and installation.
- Original error flag is set (DMU\_HF\_ERRSR.ORIER = 1<sub>B</sub>).

If both ORIG and COPY confirmation codes are an **ERRORED** value or contain uncorrectable ECC errors then:

- The UCB confirmation state is **ERRORED**
- Protection error flag is set (DMU\_HF\_ERRSR.PROER = 1<sub>B</sub>).

### **Dual UCB: Errored State or ECC Error in the UCB Confirmation Codes**

If the ORIG confirmation code is an **ERRORED** value or contains an uncorrectable ECC error then:

- The COPY confirmation code will be read to determine the UCB confirmation state and installation.

If the COPY confirmation code is an **ERRORED** value or contains an uncorrectable ECC error then:

- Protection error flag is set (DMU\_HF\_ERRSR.PROER = 1<sub>B</sub>).

### **ECC Error in the UCB Content**

If the UCB content contains an uncorrectable ECC error then:

- Protection error flag is set (DMU\_HF\_ERRSR.PROER = 1<sub>B</sub>).

### **UCB Evaluation Failure**

The device does not boot if any of the following error conditions are detected during startup:

- For single UCB: the ORIG and COPY confirmation code is **ERRORED**.
- For dual UCB: the ORIG and COPY confirmation code is **ERRORED**.
- The UCB content contains an uncorrectable ECC error.

For an errored UCB the default values are applied as shown in the Protection Configuration Register description.

### **Dual Password UCB ORIG and COPY Re-programming**

The data stored in the ORIG and COPY of a UCB pair should be identical. If there is a need to change the data then the following sequence should be followed:

- Confirm the ORIG and COPY UCB confirmation codes are **CONFIRMED**:
  - The configuration or protection installation will be installed from the ORIG UCB.
- Apply the password to **Disable Protection**. This disables protection for both ORIG and COPY UCBs.
- Erase COPY UCB - the confirmation code is erased prior to the content and treated as **ERRORED** for the installation.
- Program COPY UCB setting the confirmation code to **CONFIRMED**.
- Erase ORIG UCB - the confirmation code is erased prior to the content and treated as **ERRORED** for the installation.
  - The configuration or protection installation will be sourced from the COPY UCB.
- Program ORIG UCB setting the confirmation code to **CONFIRMED**.
  - The configuration or protection installation will be sourced from the ORIG UCB.
- Protection is re-enabled, either via command sequence or device reboot.

The ORIG is always evaluated. The above sequence ensures that the new data is confirmed in COPY before the ORIG is re-programmed. It avoids a scenario where the ORIG confirmation state is **ERRORED** and the COPY confirmation state is **UNLOCKED**.

#### 6.5.4.3.2 UCB\_BMHDx\_ORIG and UCB\_BMHDx\_COPY (x = 0-3)

Four Boot Mode Headers (BMHD) are evaluated by the SSW.

##### **UCB\_BMHDx\_ORIG and UCB\_BMHDx\_COPY (x = 0-3) Content**

The UCB content is described in the UCB chapter.

##### **UCB\_BMHDx\_ORIG (x = 0-3) Confirmation State**

The state of UCB\_BMHDx\_ORIG is indicated by DMU\_HF\_CONFIRM0.PROINBMHDxO.

##### **UCB\_BMHDx\_COPY (x = 0-3) Confirmation State**

The state of UCB\_BMHDx\_COPY is indicated by DMU\_HF\_CONFIRM0.PROINBMHDxC.

##### **Boot Mode Header Installation**

The BMHD installation is dependent on the confirmation states of UCB\_BMHDx\_ORIG and UCB\_BMHDx\_COPY. If the confirmation code of both ORIG and COPY is **ERRORED**, SSW does not evaluate the UCB.

**Table 198 Boot Mode Header 0 Installation**

<b>UCB_BMHD0_ORIG Confirmation State</b>	<b>UCB_BMHD0_COPY Confirmation State</b>	<b>Boot Mode Header Installation</b>
UNREAD	Don't Care	No evaluation.
<b>UNLOCKED</b>	Don't Care	SSW evaluates UCB_BMHD0_ORIG. Password installed from UCB_BMHD0_ORIG.
<b>CONFIRMED</b>	Don't Care	SSW evaluates UCB_BMHD0_ORIG. Password installed from UCB_BMHD0_ORIG.
<b>ERRORED</b>	<b>UNLOCKED</b>	SSW evaluates UCB_BMHD0_COPY. Password installed from UCB_BMHD0_COPY.
<b>ERRORED</b>	<b>CONFIRMED</b>	SSW evaluates UCB_BMHD0_COPY. Password installed from UCB_BMHD0_COPY.
<b>ERRORED</b>	<b>ERRORED</b>	No evaluation. No Password installed. SSW exits with error.

**Table 199 Boot Mode Header x Installation(x= 1 - 3)**

<b>UCB_BMHDx_ORIG Confirmation State</b>	<b>UCB_BMHDx_COPY Confirmation State</b>	<b>Boot Mode Header Installation</b>
UNREAD	Don't Care	No evaluation.
<b>UNLOCKED</b>	Don't Care	SSW evaluates UCB_BMHDx_ORIG.
<b>CONFIRMED</b>	Don't Care	SSW evaluates UCB_BMHDx_ORIG.
<b>ERRORED</b>	<b>UNLOCKED</b>	SSW evaluates UCB_BMHDx_COPY.

**Table 199 Boot Mode Header x Installation(x= 1 - 3) (cont'd)**

<b>UCB_BMHDx_ORIG Confirmation State</b>	<b>UCB_BMHDx_COPY Confirmation State</b>	<b>Boot Mode Header Installation</b>
<b>ERRORED</b>	<b>CONFIRMED</b>	SSW evaluates UCB_BMHDx_COPY.
<b>ERRORED</b>	<b>ERRORED</b>	No evaluation. SSW exits with error.

**BMHD Protection Disable**

If a password is installed and **Disable Protection** with matching PW is applied then

- DMU\_HF\_PROTECT.PRODISBMHD is set to 1<sub>B</sub>

If a password is not installed then protection may not be disabled.

**UCB\_BMHD0\_ORIG and UCB\_BMHD0\_COPY Access Protection**

UCB\_BMHD0\_ORIG and UCB\_BMHD0\_COPY are write protected if one of the following condition is true:

- UCB\_BMHD0 confirmation state is **CONFIRMED** and **Disable Protection** has not been activated using the password loaded from UCB\_BMHD0.
- UCB\_BMHD0 confirmation state is **ERRORED**.

UCB\_BMHD0\_ORIG and UCB\_BMHD0\_COPY content except for Password locations may be read by every on chip bus master. Password may be read by every on chip bus master only if UCB\_BMHD0 is **UNLOCKED** or **Disable Protection** has been activated.

**UCB\_BMHDx\_ORIG and UCB\_BMHDx\_COPY (x = 1-3) Access Protection**

UCB\_BMHDx\_ORIG and UCB\_BMHDx\_COPY (x = 1-3) are write protected if one of the following condition is true:

- UCB\_BMHDx confirmation state is **CONFIRMED** and **Disable Protection** has not been activated using the password loaded from UCB\_BMHD0.
- UCB\_BMHDx confirmation state is **ERRORED**.

UCB\_BMHDx\_ORIG and UCB\_BMHDx\_COPY may be read by every on chip bus master.

**6.5.4.3.3 UCB\_SSW**

UCB\_SSW contains data supplied by IFX during device production.

**UCB\_SSW Content**

The UCB content is described in the UCB chapter.

**UCB\_SSW Confirmation State**

The state of UCB\_SSW is indicated by DMU\_HF\_CONFIRM0.PROINSSW

**UCB\_SSW Content Installation**

The content installation is dependent on the ORIG and COPY confirmation states.

If ORIG confirmation code is **ERRORED** then an original error flag is set (DMU\_HF\_ERRSR.ORIER = 1<sub>B</sub>).

**UCB\_SSW Access Protection**

UCB\_SSW is read only and can be read by every on chip bus master.

#### 6.5.4.3.4 UCB\_USER

UCB\_USER contains data supplied by IFX during device production.

##### UCB\_USER Content

The UCB content is described in the UCB chapter.

##### UCB\_USER Unique Identifier

The Unique Identifier (UID) consists of 128 bits starting at UCB\_USER address offset 000<sub>H</sub>. It contains the following information:

- Lot number: “KKJWWNNNN” with sub-lot number “xy”.
- Date code: day, month, year.
- X and Y coordinate on the wafer.
- Wafer number.
- Constant identifiers for IFX, fab, Unique Identifier format.

The 6-bit characters 00<sub>H</sub> to 23<sub>H</sub> encode the digits and letters [0-9, A-Z].

**Table 200 Unique Identifier**

Bits	Type	Content
[7:0]	8-bit unsigned int	manufacturer ID (40 <sub>H</sub> )
[9:8]		unused
[14:10]	6-bit unsigned int	split lot char 2[bits 5:1]
[22:15]	8-bit unsigned int	Y coordinate on wafer
[30:23]	8-bit unsigned int	X coordinate on wafer
[31:31]		unused
[43:38]	6-bit unsigned int	split log char 1 [bit 5..0]
[44:44]	1-bit unsigned int	split lot char 2 [bit 0]
[45:45]	1-bit unsigned int	indicator for wafer test or blind assembly (0=wafer, 1=blind assembly)
[49:46]		unused
[59:50]	10-bit unsigned int	counter for serial lot numbers (001-999 <sub>D</sub> ) - NNN
[65:60]	6-bit unsigned int	Infineon logistic week during lot creation - WW
[69:66]	4-bit unsigned int	code number of Infineon financial year (last digit) - J
[74:70]	5-bit unsigned int	Facility code “19” = 7 (FabID) - KK
[79:75]	5-bit unsigned int	day of date code (range 1 to 31 <sub>D</sub> )
[83:80]	4-bit unsigned int	month of date code (range 1 to 12 <sub>D</sub> )
[91:84]	8-bit unsigned int	year of date code (add 2000 <sub>D</sub> )
[95:92]		constant 0100 <sub>B</sub> identifying the supplier
[127:96]		unused

##### UCB\_USER Confirmation State

The state of UCB\_USER is indicated by DMU\_HF\_CONFIRM0.PROINUSER.

### UCB\_USER Content Installation

The content installation is dependent on the ORIG and COPY confirmation states.

If ORIG confirmation code is **ERRORED** then an original error flag is set (DMU\_HF\_ERRSR.ORIER = 1<sub>B</sub>).

### UCB\_USER Access Protection

UCB\_USER is read only and can be read by every on chip bus master.

### 6.5.4.3.5 UCB\_TEST

UCB\_TEST contains test information supplied by IFX during device production.

#### UCB\_TEST Content

The UCB content is described in the Test specification.

#### UCB\_TEST Test Pass Marker

The 4 bytes starting at UCB\_TEST address offset 180<sub>H</sub> contain the “Test Pass Marker”. During test the Test Pass Marker is set to FFFFFFFF<sub>H</sub>. When all tests have passed the Test Pass Marker is programmed to 80658383<sub>H</sub>. A production device with a different value may be discarded.

#### UCB\_TEST Confirmation State

The state of UCB\_TEST is indicated by DMU\_HF\_CONFIRM0.PROINTEST.

#### UCB\_TEST Content Installation

The content installation is dependent on the ORIG and COPY confirmation states.

If ORIG confirmation code is **ERRORED** then an original error flag is set (DMU\_HF\_ERRSR.ORIER = 1<sub>B</sub>).

### UCB\_TEST Access Protection

UCB\_TEST is read only and can be read by every on chip bus master.

### 6.5.4.3.6 UCB\_HSMCFG

This UCB contains data supplied by Infineon during device production for the exclusive use of the HSM module.

#### UCB\_HSMCFG Content

If the HSM is configured then the HSM DLT routine has programmed the HSM keys and configuration data.

**Table 201 UCB\_HSMCFG Content**

Offset	Content	Range	Description
000 <sub>H</sub>	HSM Configuration		Defined in HSM ITS. It contains: <ul style="list-style-type: none"><li>• 128-bit Hash of the data.</li><li>• 128-bit AES key 0.</li><li>• 128-bit AES key 1.</li><li>• TRNG configuration</li><li>• Versioning.</li></ul>
1F0 <sub>H</sub>	ORIG Confirmation	4 Bytes	32-bit confirmation code.
1F8 <sub>H</sub>	COPY Confirmation	4 Bytes	32-bit confirmation code.

If the HSM is not configured then the UCB\_HSMCFG content should be programmed with all zeros.

#### **UCB\_HSMCFG Confirmation State**

The confirmed state of UCB\_HSMCFG is indicated by DMU\_HF\_CONFIRM0.PROINHSMCFG.

If ORIG confirmation code is **ERRORED** then an original error flag is set (DMU\_HF\_ERRSR.ORIER = 1<sub>B</sub>).

#### **UCB\_HSMCFG Write Access Protection**

UCB\_HSMCFG may be programmed and erased by all masters when all of the following conditions are true:

- UCB\_HSMCFG confirmation state is **UNLOCKED**.

#### **UCB\_HSMCFG Read Access Protection**

UCB\_HSMCFG may be read by all on chip masters if the following condition is true:

- UCB\_HSMCFG confirmation state is **UNLOCKED**.

UCB\_HSMCFG may be read only by the HSM on chip master if the following condition is true:

- UCB\_HSMCFG ORIG confirmation state is **CONFIRMED** or **ERRORED**.

The HSM can lock read access completely until the next application reset with DMU\_SF\_CONTROL.LCKHSMUCB. This locking is performed as part of the HSM startup by its firmware.

If HSM debug mode is enabled then the Cerberus has the same access rights as HSM else the Cerberus is treated as any other on chip bus master.

#### **6.5.4.3.7 UCB\_REDSEC**

This UCB contains the redundancy information for all the flash banks.

#### **UCB\_REDSEC**

The UCB content is described in the UCB chapter.

#### **UCB\_REDSEC Confirmation**

The confirmation state of UCB\_REDSEC is indicated by DMU\_HF\_CONFIRM0.PROINREDSEC.

If ORIG confirmation code is **ERRORED** then an original error flag is set (DMU\_HF\_ERRSR.ORIER = 1<sub>B</sub>).

#### **UCB\_REDSEC Access Protection**

UCB\_REDSEC is read only and can be read by every on chip bus master.

#### **6.5.4.3.8 UCB\_PFLASH\_ORIG and UCB\_PFLASH\_COPY**

Password protection of PFLASH banks is stored in UCB\_PFLASH\_ORIG and UCB\_PFLASH\_COPY.

#### **UCB\_PFLASH\_ORIG and UCB\_PFLASH\_COPY Content**

The UCB content is described in the UCB chapter.

#### **UCB\_PFLASH\_ORIG Confirmation State**

The confirmation state of UCB\_PFLASH\_ORIG is indicated by DMU\_HF\_CONFIRM1.PROINPO.

#### **UCB\_PFLASH\_COPY Confirmation State**

The confirmation state of UCB\_PFLASH\_COPY is indicated by DMU\_HF\_CONFIRM1.PROINPC.

## Program Flash Protection and Password Installation

The protection and password installation is dependent on the confirmation states of UCB\_PFLASH\_ORIG and UCB\_PFLASH\_COPY (at [Table 202](#)).

**Table 202 Program Flash Protection and Password Installation**

UCB_PFLASH_ORIG Confirmation State	UCB_PFLASH_COPY Confirmation State	Protection and Password Installation
UNREAD	Don't Care	Reset value. No password installed.
<b>UNLOCKED</b>	Don't Care	Protection installed from UCB_PFLASH_ORIG. Password installed from UCB_PFLASH_ORIG.
<b>CONFIRMED</b>	Don't Care	Protection installed from UCB_PFLASH_ORIG. Password installed from UCB_PFLASH_ORIG.
<b>ERRORED</b>	<b>UNLOCKED</b>	Protection installed from UCB_PFLASH_COPY. Password installed from UCB_PFLASH_COPY.
<b>ERRORED</b>	<b>CONFIRMED</b>	Protection installed from UCB_PFLASH_COPY. Password installed from UCB_PFLASH_COPY.
<b>ERRORED</b>	<b>ERRORED</b>	Default protection installed. No password installed. SSW exits with error.

## Program Flash Protection Disable

If a password is installed and **Disable Protection** with matching PW is applied then

- DMU\_HF\_PROTECT.PRODISP is set to 1<sub>B</sub>

If a password is not installed then protection may not be disabled.

## UCB\_PFLASH\_ORIG and UCB\_PFLASH\_COPY Access Protection

UCB\_PFLASH\_ORIG and UCB\_PFLASH\_COPY are read and write protected if one of the following conditions is true:

- The UCB\_PFLASH confirmation state is **CONFIRMED** and **Disable Protection** has not been activated.
- The UCB\_PFLASH confirmation state is **ERRORED**.

### 6.5.4.3.9 UCB\_DFLASH\_ORIG and UCB\_DFLASH\_COPY

Password protection of the DFLASH is stored in UCB\_DFLASH\_ORIG and UCB\_DFLASH\_COPY. The UCB is also used for configuring the memory initialization and contains an oscillator configuration.

#### UCB\_DFLASH\_ORIG and UCB\_DFLASH\_COPY Content

The UCB content is described in the UCB chapter.

#### UCB\_DFLASH\_ORIG Confirmation

The confirmation state of UCB\_DFLASH\_ORIG is indicated by DMU\_HF\_CONFIRM1.PROINDO.

#### UCB\_DFLASH\_COPY Confirmation

The confirmation state of UCB\_DFLASH\_COPY is indicated by DMU\_HF\_CONFIRM1.PROINDC.

### Data Flash Protection and Password Installation

The protection and password installation is dependent on the confirmation states of UCB\_DFLASH\_ORIG and UCB\_DFLASH\_COPY (at [Table 203](#)).

**Table 203 Data Flash Protection and Password Installation**

UCB_DFLASH_ORIG Confirmation State	UCB_DFLASH_COPY Confirmation State	Protection and Password Installation
UNREAD	Don't Care	Reset value. No password installed.
<b>UNLOCKED</b>	Don't Care	Protection installed from UCB_DFLASH_ORIG. Password installed from UCB_DFLASH_ORIG.
<b>CONFIRMED</b>	Don't Care	Protection installed from UCB_DFLASH_ORIG. Password installed from UCB_DFLASH_ORIG.
<b>ERRORED</b>	<b>UNLOCKED</b>	Protection installed from UCB_DFLASH_COPY. Password installed from UCB_DFLASH_COPY.
<b>ERRORED</b>	<b>CONFIRMED</b>	Protection installed from UCB_DFLASH_COPY. Password installed from UCB_DFLASH_COPY.
<b>ERRORED</b>	<b>ERRORED</b>	Default protection installed. No password installed. SSW exits with error.

### Data Flash Protection Disable

If a password is installed and **Disable Protection** with matching PW is applied then

- DMU\_HF\_PROTECT.PRODISD is set to 1<sub>B</sub>

If a password is not installed then protection may not be disabled.

### UCB\_DFLASH\_ORIG and UCB\_DFLASH\_COPY Access Protection

UCB\_DFLASH\_ORIG and UCB\_DFLASH\_COPY are read and write protected if one of the following conditions is true:

- The UCB\_DFLASH confirmation state is **CONFIRMED** and **Disable Protection** has not been activated.
- The UCB\_DFLASH confirmation state is **ERRORED**.

### 6.5.4.3.10 UCB\_DBG\_ORIG and UCB\_DBG\_COPY

This UCB configures the password protection for the debug interface.

#### UCB\_DBG\_ORIG and UCB\_DBG\_COPY Content

The UCB content is described in the UCB chapter.

#### UCB\_DBG\_ORIG Confirmation

The confirmation state of UCB\_DBG\_ORIG is indicated by DMU\_HF\_CONFIRM1.PROINDBGO.

#### UCB\_DBG\_COPY Confirmation

The confirmation state of UCB\_DBG\_COPY is indicated by DMU\_HF\_CONFIRM1.PROINDBGC.

### Debug Protection and Password Installation

The protection and password installation is dependent on the confirmation states of UCB\_DBG\_ORIG and UCB\_DBG\_COPY (at [Table 204](#)).

**Table 204 Debug Protection and Password Installation**

UCB_DBG_ORIG Confirmation State	UCB_DBG_COPY Confirmation State	Protection and Password Installation
UNREAD	Don't Care	Reset value. No password installed.
UNLOCKED	Don't Care	Protection installed from UCB_DBG_ORIG. Password installed from UCB_DBG_ORIG.
CONFIRMED	Don't Care	Protection installed from UCB_DBG_ORIG. Password installed from UCB_DBG_ORIG.
ERRORED	UNLOCKED	Protection installed from UCB_DBG_COPY. Password installed from UCB_DBG_COPY.
ERRORED	CONFIRMED	Protection installed from UCB_DBG_COPY. Password installed from UCB_DBG_COPY.
ERRORED	ERRORED	Default protection installed. No password installed. SSW exits with error.

### Debug Protection Disable

If a password is installed and **Disable Protection** with matching PW is applied then

- DMU\_HF\_PROTECT.PRODISDBG is set to 1<sub>B</sub>

If a password is not installed then protection may not be disabled.

### UCB\_DBG\_ORIG and UCB\_DBG\_COPY Access Protection

UCB\_DBG\_ORIG and UCB\_DBG\_COPY are read and write protected if one of the following conditions is true:

- The UCB\_DBG confirmation state is **CONFIRMED** and **Disable Protection** has not been activated.
- The UCB\_DBG confirmation state is **ERRORED**.
- Debug entry is destructive:
  - DMU\_SP\_PROCONHSMCFG.DESTDBG = 11<sub>B</sub> and (FDEST = 0<sub>B</sub> or SCU\_STCON.STP = 1<sub>B</sub>)
- Debug entered:
  - DMU\_HF\_PROCONDDBG.EDM = 11<sub>B</sub>

### 6.5.4.3.11 UCB\_HSM\_ORIG and UCB\_HSM\_COPY

UCB\_HSM\_ORIG and UCB\_HSM\_COPY configures the HSM interface protection and the content is only used in devices with activated HSM.

If HSM debug mode is enabled then the Cerberus has the same access rights as HSM else the Cerberus is treated as any other on chip bus master.

### UCB\_HSM\_ORIG and UCB\_HSM\_COPY Content

The UCB content is described in the UCB chapter.

### UCB\_HSM\_ORIG Confirmation

The confirmation state of UCB\_HSM\_ORIG is indicated by DMU\_HF\_CONFIRM1.PROINHSMO.

### UCB\_HSM\_COPY Confirmation

The confirmation state of UCB\_HSM\_COPY is indicated by DMU\_HF\_CONFIRM1.PROINHSMC.

### HSM Configuration Installation

The protection installation is dependent on the confirmation states of UCB\_HSM\_ORIG and UCB\_HSM\_COPY (at [Table 205](#)).

**Table 205 HSM Configuration Installation**

UCB_HSM_ORIG Confirmation State	UCB_HSM_COPY Confirmation State	Protection Installation
UNREAD	Don't Care	Reset value.
<b>UNLOCKED</b>	Don't Care	Protection installed from UCB_HSM_ORIG.
<b>CONFIRMED</b>	Don't Care	Protection installed from UCB_HSM_ORIG.
<b>ERRORED</b>	<b>UNLOCKED</b>	Protection installed from UCB_HSM_COPY.
<b>ERRORED</b>	<b>CONFIRMED</b>	Protection installed from UCB_HSM_COPY.
<b>ERRORED</b>	<b>ERRORED</b>	Default protection installed. SSW exits with error.

### UCB\_HSM\_ORIG and UCB\_HSM\_COPY Access Protection

UCB\_HSM\_ORIG and UCB\_HSM\_COPY are protected from programming and erasing by non-HSM bus masters if one of the following conditions is true:

- UCB\_HSM confirmation state is **CONFIRMED**.
- UCB\_HSM confirmation state is **ERRORED**.

UCB\_HSM\_ORIG and UCB\_HSM\_COPY may be read by every on chip bus master.

### 6.5.4.3.12 UCB\_HSMCOTP0/1\_ORIG and UCB\_HSMCOTP0/1\_COPY

HSMCOTP protection configures the HSM\_exclusive and OTP protection for the dedicated HSM flash sectors. It offers the possibility to add this type of protection to Flash sectors incrementally from two configuration sets:

- HSMCOTP0 configuration set derived from UCB\_HSMCOTP0\_ORIG and UCB\_HSMCOTP0\_COPY.
- HSMCOTP1 configuration set derived from UCB\_HSMCOTP1\_ORIG and UCB\_HSMCOTP1\_COPY.

### HSMCOTP Use Case

All masters can configure the HSMCOTP protection when both HSMCOTP0 and HSMCOTP1 configuration sets are **UNLOCKED**. If the HSMCOTP0 configuration set is **CONFIRMED** then only the HSM is allowed to program the HSMCOTP1 configuration set. If HSMCOTP1 is also **CONFIRMED** then the complete HSMCOTP configuration set is OTP protected.

### UCB\_HSMCOTP0/1\_ORIG and UCB\_HSMCOTP0/1\_COPY Content

The UCB content is described in the UCB chapter.

### **UCB\_HSMCOTP0\_ORIG Confirmation State**

The confirmation state of UCB\_HSMCOTP0\_ORIG is indicated by DMU\_HF\_CONFIRM1.PROINHSMCOTP00.

### **UCB\_HSMCOTP0\_COPY Confirmation State**

The confirmation state of UCB\_HSMCOTP0\_COPY is indicated by DMU\_HF\_CONFIRM1.PROINHSMCOTP0C.

### **UCB\_HSMCOTP1\_ORIG Confirmation State**

The confirmation state of UCB\_HSMCOTP1\_ORIG is indicated by DMU\_HF\_CONFIRM1.PROINHSMCOTP10.

### **UCB\_HSMCOTP1\_COPY Confirmation State**

The confirmation state of UCB\_HSMCOTP1\_COPY is indicated by DMU\_HF\_CONFIRM1.PROINHSMCOTP1C.

### **HSMCOTP0 Confirmation State**

The confirmation state of HSMCOTP0 is the confirmation state of UCB\_HSMCOTP0\_ORIG if **UNLOCKED** or **CONFIRMED**, else it is that of UCB\_HSMCOTP0\_COPY.

### **HSMCOTP1 Confirmation State**

The confirmation state of HSMCOTP1 is the confirmation state of UCB\_HSMCOTP1\_ORIG if **UNLOCKED** or **CONFIRMED**, else it is that of UCB\_HSMCOTP1\_COPY.

### **HSMCOTP Protection Configuration Installation**

The HSMCOTP protection configuration is installed as follows:

- Initial Protection
  - If the HSMCOTP0 confirmation state is **UNLOCKED** or **CONFIRMED** then the content of the DMU\_SP\_PROCONHSMCFG, DMU\_SP\_PROCONHSMCBS, DMU\_SP\_PROCONHSMCX0, DMU\_SP\_PROCONHSMCX1, DMU\_SP\_PROCONHSMCOTP0 and DMU\_SP\_PROCONHSMCOTP1 registers are initialized to HSMCOTP0.
- Subsequent Protection
  - If the HSMCOTP1 confirmation state is **CONFIRMED** then the HSMCOTP1 is OR'ed to the initial value.
  - If the confirmation state of either HSMCOTP0 or HSMCOTP1 is **ERRORED**, default protection is installed.

### **UCB\_HSMCOTP0/1\_ORIG and UCB\_HSMCOTP0/1\_COPY Erase Protection**

UCB\_HSMCOTP0\_ORIG, UCB\_HSMCOTP0\_COPY, UCB\_HSMCOTP1\_ORIG and UCB\_HSMCOTP1\_COPY are protected from erasing when the confirmation state of at least one of the HSMCOTP configuration sets is **CONFIRMED** or **ERRORED**.

### **UCB\_HSMCOTP0\_ORIG and UCB\_HSMCOTP0\_COPY Program Protection**

UCB\_HSMCOTP0\_ORIG and UCB\_HSMCOTP0\_COPY are protected from programming for all masters when one of the following conditions is true:

- HSMCOTP0 configuration set confirmation state is **CONFIRMED** or **ERRORED**.
- HSMCOTP1 configuration set confirmation state is **ERRORED**.

### **UCB\_HSMCOTP1\_ORIG and UCB\_HSMCOTP1\_COPY Program Protection**

UCB\_HSMCOTP1\_ORIG and UCB\_HSMCOTP1\_COPY are protected from programming for all masters except HSM when one of the following conditions is true:

- HSMCOTP0 configuration set confirmation state is **CONFIRMED**.

UCB\_HSMCOTP1\_ORIG and UCB\_HSMCOTP1\_COPY are protected from programming for all masters including the HSM when one of the following conditions is true:

- HSMCOTP0 configuration set confirmation state is **ERRORED**.
- HSMCOTP1 configuration set confirmation state is **CONFIRMED** or **ERRORED**.

#### **UCB\_HSMCOTP0/1\_ORIG and UCB\_HSMCOTP0/1\_COPY Read Protection**

UCB\_HSMCOTP0/1\_ORIG and UCB\_HSMCOTP0/1\_COPY are readable by all on chip bus masters.

#### **6.5.4.3.13 UCB\_ECPRIORIG and UCB\_ECPRIOCOPY**

UCB\_ECPRIORIG and UCB\_ECPRIOCOPY is used to set priority to the PFLASH logical sectors for use in recording of erase operations in the Erase Counter area.

#### **UCB\_HSMCOTP0/1\_ORIG and UCB\_HSMCOTP0/1\_COPY Content**

The UCB content is described in the UCB chapter.

#### **UCB\_ECPRIORIG Confirmation State**

The confirmation state of UCB\_ECPRIORIG is indicated by DMU\_HF\_CONFIRM1.PROINECPRIORO.

#### **UCB\_ECPRIOCOPY Confirmation State**

The confirmation state of UCB\_ECPRIOCOPY is indicated by DMU\_HF\_CONFIRM1.PROINECPRIOC.

#### **ECPRIOR Confirmation State**

The confirmation state of ECPRIOR is the confirmation state of UCB\_ECPRIORIG if **UNLOCKED** or **CONFIRMED**, else it is that of UCB\_ECPRIOCOPY.

#### **Priority and Password Installation**

The priority information and password installation is dependent on the confirmation states of UCB\_ECPRIORIG and UCB\_ECPRIOCOPY (at [Table 204](#)).

**Table 206 Priority and Password Installation**

<b>UCB_ECPRIORIG Confirmation State</b>	<b>UCB_ECPRIOCOPY Confirmation State</b>	<b>Protection and Password Installation</b>
UNREAD	Don't Care	Reset value. No password installed.
<b>UNLOCKED</b>	Don't Care	Priority installed from UCB_ECPRIORIG. Password installed from UCB_ECPRIORIG.
<b>CONFIRMED</b>	Don't Care	Priority installed from UCB_ECPRIORIG. Password installed from UCB_ECPRIORIG.
<b>ERRORED</b>	<b>UNLOCKED</b>	Priority installed from UCB_ECPRIOCOPY. Password installed from UCB_ECPRIOCOPY.

**Table 206 Priority and Password Installation (cont'd)**

UCB_ECPRIOR_ORIG Confirmation State	UCB_ECPRIOR_COPY Confirmation State	Protection and Password Installation
ERRORED	CONFIRMED	Priority installed from UCB_ECPRIOR_COPY. Password installed from UCB_ECPRIOR_COPY.
ERRORED	ERRORED	Default Priority installed. No password installed. SSW exits with error.

### Program Flash Protection Disable

If a password is installed and **Disable Protection** with matching PW is applied then

- DMU\_HF\_PROTECT.PRODISEC is set to 1<sub>B</sub>

If a password is not installed then protection may not be disabled.

### UCB\_ECPRIOR\_ORIG and UCB\_ECPRIOR\_COPY Access Protection

UCB\_ECPRIOR\_ORIG and UCB\_ECPRIOR\_COPY are read and write protected if one of the following conditions is true:

- The UCB\_ECPRIOR confirmation state is **CONFIRMED** and **Disable Protection** has not been activated.
- The UCB\_ECPRIOR confirmation state is **ERRORED**.

### 6.5.4.3.14 UCB\_SWAP\_ORIG and UCB\_SWAP\_COPY

UCB\_SWAP is evaluated by the SSW to determine the Pflashes used by the running application (referred to as 'SWAP' in this chapter). Refer to "Software Update Over the Air (SOTA) section of the User Manual for more details.

#### UCB\_SWAP\_ORIG and UCB\_SWAP\_COPY Content

The UCB content is described in the UCB chapter.

#### UCB\_SWAP\_ORIG Confirmation State

The state of UCB\_SWAP\_ORIG is indicated by DMU\_HF\_CONFIRM1.PROINSWAPO.

#### UCB\_SWAP\_COPY Confirmation State

The state of UCB\_SWAP\_COPY is indicated by DMU\_HF\_CONFIRM1.PROINSWAPC.

#### SWAP Installation

The SWAP installation is dependent on the confirmation states of UCB\_SWAP\_ORIG and UCB\_SWAP\_COPY. If the confirmation code of both ORIG and COPY is **ERRORED**, SSW does not evaluate the UCB.

**Table 207 SWAP Installation**

<b>UCB_SWAP_O RIG Confirmation State</b>	<b>UCB_SWAP_ COPY Confirmation State</b>	<b>SWAP confirmation</b>	<b>SWAP marker</b>	<b>SWAP installation</b>
<b>UNLOCKED/C ONFIRMED</b>	Don't Care	Valid in ORIG	Valid in ORIG	SWAP installed from ORIG after SSW evaluates the SWAP confirmation and SWAP marker.
<b>UNLOCKED/C ONFIRMED</b>	Don't Care	Invalid in ORIG	Don't care	SSW exits with error
<b>UNLOCKED/C ONFIRMED</b>	Don't Care	Don't care	Invalid in ORIG	SSW exits with error
<b>ERRORED</b>	<b>UNLOCKED/C ONFIRMED</b>	Valid in COPY	Valid in COPY	SWAP installed from COPY after SSW evaluates the SWAP confirmation and SWAP marker.
<b>ERRORED</b>	<b>UNLOCKED/C ONFIRMED</b>	Invalid in COPY	Don't care	SSW exits with error
<b>ERRORED</b>	<b>UNLOCKED/C ONFIRMED</b>	Don't care	Invalid in COPY	SSW exits with error
<b>ERRORED</b>	<b>ERRORED</b>	Don't care	Don't care	SSW exits with error

**SWAP Protection Disable**

If a password is installed and **Disable Protection** with matching PW is applied then

- DMU\_HF\_PROTECT.PRODISSWAP is set to 1<sub>B</sub>

If a password is not installed then protection may not be disabled.

**UCB\_SWAP\_ORIG and UCB\_SWAP\_COPY Access Protection**

UCB\_SWAP\_ORIG and UCB\_SWAP\_COPY are write protected if one of the following condition is true:

- UCB\_SWAP confirmation state is **CONFIRMED** and **Disable Protection** has not been activated using the password loaded from UCB\_SWAP.
- UCB\_SWAP confirmation state is **ERRORED**.

UCB\_SWAP\_ORIG and UCB\_SWAP\_COPY content except for Password locations may be read by every on chip bus master. Password may be read by every on chip bus master only if UCB\_SWAP is **UNLOCKED** or **Disable Protection** has been activated.

**6.5.4.3.15 UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY (y = 0-7)**

UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY configure sets of one-time programmable “OTP” and write-page once “WOP” protection. Each OTP and WOP configuration set supports the incremental addition of OTP and WOP protection to the PFLASH sectors. UCB\_OTP also contains the **HF\_PROCONT** register that is used to configure Tuning Protection and Software Update Over the Air (SOTA) in the device.

**UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY Content**

The UCB content is described in the UCB chapter.

**Note:** *The memory size of the UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY is fixed but the memory content will vary with the number of PFLASH banks.*

### **UCB\_OTPy\_ORIG Confirmation State**

The state of UCB\_OTPy\_ORIG is indicated by DMU\_HF\_CONFIRM2.PROINOTPyO.

### **UCB\_OTPy\_COPY Confirmation State**

The state of UCB\_OTPy\_COPY is indicated by DMU\_HF\_CONFIRM2.PROINOTPyC.

### **OTP Protection Installation**

The confirmation state of the complete set of the UCB\_OTP sets may be represented by PROINOTP.

PROINOTP = 1<sub>B</sub> when the following condition is true:

- For y = 0 - 7, if any one of the UCB\_OTPy confirmation states is **CONFIRMED** or **ERRORED**.

#### Initial Protection

- If UCB\_OTP0\_ORIG is **UNLOCKED** or **CONFIRMED** then
  - PROCONTP, PROCONOTP and PROCONWOP registers are initialized to the UCB\_OTP0\_ORIG configuration set.
- Else if UCB\_OTP0\_COPY is **UNLOCKED** or **CONFIRMED** then
  - PROCONTP, PROCONOTP and PROCONWOP registers are initialized to the UCB\_OTP0\_COPY configuration set.

#### Subsequent Protection

- If UCB\_OTP1\_ORIG is **CONFIRMED** then
  - PROCONTP, PROCONOTP and PROCONWOP registers are programmed to UCB\_OTP0 OR'ed UCB\_OTP1\_ORIG protection configuration sets.
- Else if UCB\_OTP1\_ORIG is **ERRORED** and UCB\_OTP1\_COPY is **CONFIRMED** then
  - PROCONTP, PROCONOTP and PROCONWOP registers are programmed to UCB\_OTP0 OR'ed UCB\_OTP1\_COPY protection configuration sets.
- If subsequent UCB\_OTPy are **CONFIRMED** then
  - PROCONTP, PROCONOTP and PROCONWOP registers are programmed to OR'ed protection configuration sets of all preceding confirmed UCBs.

#### Error

- If (for common y) UCB\_OTPy\_ORIG is **ERRORED** and UCB\_OTPy\_COPY is **ERRORED** then
  - All SxROM bits and SxWOP bits are activated for the complete OTP configuration set.

### **UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY Erase Access Protection**

UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY (for all y) are protected from erasing if PROINOTP = 1<sub>B</sub>

### **UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY Program Access Protection**

If for any y UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY the following condition is true:

- UCB\_OTPy\_ORIG confirmation state is **UNLOCKED**.

then UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY may be programmed.

### **UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY Read Access Protection**

UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY are readable.

#### **6.5.4.3.16 Spare UCB**

A number of UCBs are currently not used (see UCB chapter).

##### **Spare UCB Content**

The content is undefined.

##### **Spare UCB Confirmation**

The confirmation state of spare UCBs is not supported.

##### **Spare UCB Access Protection**

Spare UCBs are read only and can be read by every on chip bus master.

#### **6.5.4.4 System Wide Effects of Flash Protection**

An active Flash read protection needs to be respected in the complete system.

The startup software “SSW” checks if the HSM is available.

If yes the HSM module is booted. During its boot process it can lock the device debug interface. This interface can be locked either by HSM (see below **HSM Booting**) or by setting OSTATE.IF\_LCK.

Additionally the customer can configure in UCB\_DBG\_ORIG/UCB\_DBG\_COPY that OCDS or the debug interface are disabled.

This results in the following lock conditions:

- OCDS disabled: DMU\_HF\_PROCONDBG.OCDSDIS and not DMU\_HF\_PROTECT.PRODISDBG.
- Debug interface locked: OSTATE.IF\_LCK<sup>1)</sup> or (HSM lock input) or (DMU\_HF\_PROCONDBG.DBGIFLCK and not DMU\_HF\_PROTECT.PRODISDBG).

The SSW performs the following operations:

- The SSW leaves the debug interface locked (OSTATE.IF\_LCK stays 1) if any Flash read protection is configured (DMU\_HF\_PROCONPF.RPRO or DMU\_HF\_PROCONPF.RPRO are 1).
- If the selected boot mode executes from internal PFLASH:
  - The SSW clears the DMU\_HF\_CONTROL.DDFP and DMU\_HF\_CONTROL.DDFD
- If the selected boot mode does not execute from internal PFLASH:
  - The SSW sets the DMU\_HF\_CONTROL.DDFP and DMU\_HF\_CONTROL.DDFD if their corresponding read protection is configured in DMU\_HF\_PROCONPF.RPRO or DMU\_HF\_PROCONDF.RPRO.

Full Flash analysis of an FAR device is only possible when the customer has removed all installed protections or delivers the necessary passwords with the device. As the removal of an OTP protection is not possible the OTP protection inevitably limits analysis capabilities.

For devices supporting HSM the FAR capabilities are generally limited to protect HSM configuration data.

#### **6.5.4.4.1 HSM Booting**

The SSW boots the HSM module (i.e. HSM executes its firmware) when the field DMU\_SP\_PROCONHSMCFG.HSMBOOTEN is set (i.e. HSM boot code is installed in the HSM code sectors).

First the HSM firmware checks DMU\_SP\_PROCONHSM.HSMDBGDIS. If this is cleared it enables HSM debug.

After that the HSM firmware checks DMU\_SP\_PROCONHSM.DBGIFLCK. If this is cleared it releases its lock of the system debug interface.

1) Only for illustration, this bit and its or-combination with the following signals is part of the OCDS not the DMU.

After execution of the HSM firmware the HSM executes from the HSM code sectors. Depending on DMU\_SP\_PROCONHSMCFG.SSWWAIT the SSW waits for an acknowledge from the HSM before leaving the Firmware.

#### 6.5.4.4.2 Destructive Debug Entry

As described before the debug interface can be opened by supplying the correct password to UCB\_DBG\_ORIG or UCB\_DBG\_COPY

This is under the condition that HSM doesn't lock this interface.

A special destructive debug entry can be configured with DMU\_SP\_PROCONHSMCFG.DESTDBG. When this is configured as "destructive" only the SSW can accept the UCB\_DBG\_ORIG or UCB\_DBG\_COPY password via debug interface (see BROM chapter). Additionally, read and write accesses are permitted to UCB\_DBG\_ORIG and UCB\_DBG\_COPY only if the device pin "FDEST" is logic '1' and the field DMU\_HF\_PROCONDDBG.EDM is "debug not entered". When all these conditions match the field DMU\_HF\_PROCONDDBG.EDM is programmed by the SSW to "debug entered" before opening the debug interface. When EDM is "debug entered", then automatically any communication via CAN or FlexRay is blocked.

## 6.5.5 Revision History

**Table 208 Revision History**

Reference	Change to Previous Version	Comment
<b>V2.0.9</b>		
Page 90	<b>HF_PWAIT</b> - Removed errant p in the RFLASH, RECC, CFLASH and CECC bit field descriptions.	
<b>V2.0.10</b>		
Page 31	<b>Write Burst Once</b> - Only an EVER is raised if the sequence is performed on an unerased area, not a PVER and EVER. Also changed wording to make clear that pages are effected, not a single page.	
Page 92	<b>HF_DWAIT</b> - In RECC description, changed PFLASH to DFLASH - header was correct.	
<b>V2.0.11</b>		
Page 63	Updated register <b>HF_STATUS</b> .	

## 6.6 Program Flash Interface (PFI)

### 6.6.1 Overview

The PFI (in [Figure 64](#)) contains two distinct signal paths that drive a PFI to CPU point-to-point connection:

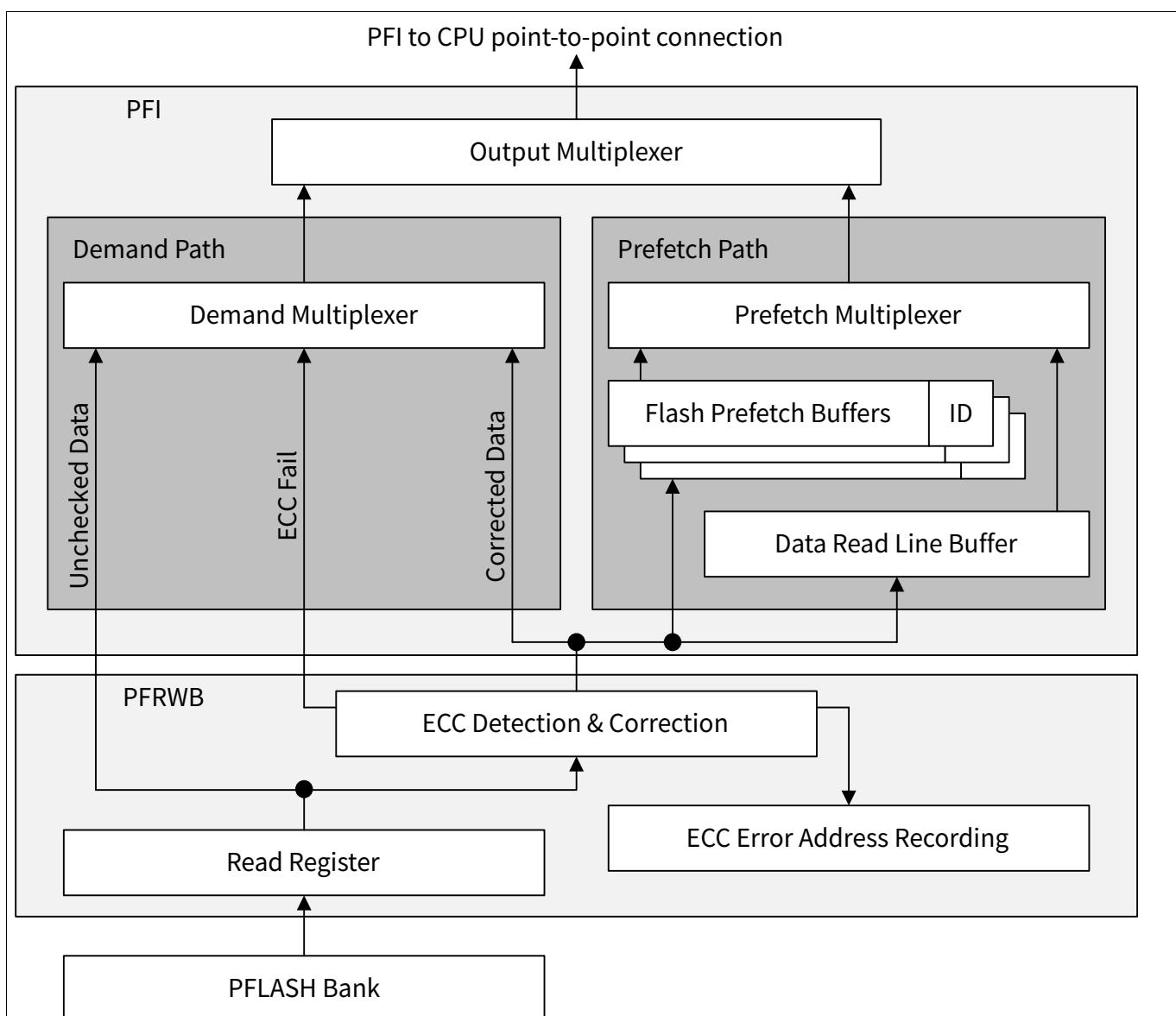
- Demand path to support early transmission of data.
- Prefetch path to support speculative fetch of data.

The PFLASH bank read data is available as follows:

- Uncorrected data is available at the read register output. ECC fail detects if the read data contains an ECC error.
- Corrected data is available after the ECC decoder has checked the data integrity.

ECC checksums protect the integrity of data as follows:

- Data programmed in the PFLASH bank is stored with an ECC checksum extension.
- Data transported across the PFI to CPU point-to-point connection is protected by an ECC sideband signal.



**Figure 64 Block Diagram of the PFI module.**

## 6.6.2 Functional Description

### 6.6.2.1 Demand Path

Local CPU PMBI program code fetches, local CPU DMBI Non Safe load data constants, local CPU DMBI Safe load data constants and SRI accesses requested by remote bus masters which cannot be serviced by a prefetch access (see [Chapter 6.6.2.3](#)) are serviced by a demand access direct to the PFLASH bank. PFLASH read data is transmitted after a number of read cycles determined by the configuration. If the access is a block transfer then the critical response (double word) must be transmitted first. If the block transfer is a BTR4 originating from the PMBI or the DMBI then the critical double word may be transmitted early using the uncorrected data. If an ECC failure is detected then the critical response is re-transmitted using corrected data.

### 6.6.2.2 Data Read Line Buffer (DRLB)

In order to improve the efficiency of SDTD and BTR2 accesses a DRLB stores the PFLASH page data and address. If a subsequent access hits the DRLB then the response is serviced by the DRLB.

### 6.6.2.3 Flash Prefetch Buffer (FPB)

The prefetch path instantiates prefetch buffer(s) to speculatively fetch data. If a BTR4 access hits the FPB then the response is serviced by the FPB.

#### FPB Configuration

The local CPU contains a set of FLASHCON registers for configuring its local PFI. Each PFI instantiates five FPBs. An FPB may be assigned to an on chip bus master (see [Table 209](#)) by software programming the 6-bit master tag identification number to the appropriate fields of the CPU\_FLASHCON0 register.

**Table 209 FPB Assignment**

FPB	Description
[0]	Permanently assigned to the local CPU PMBI master tag identification number.
[4:1]	Assigned to an on chip bus master by software programming the CPU_FLASHCON0 register with the appropriate 6-bit master tag identification number.  <i>Note:</i> To reduce power consumption prefetching from an individual FPB may be disabled by software assigning the FPB to the reserved 6-bit master tag identification number.

Performance may be optimized by assigning a maximum of two FPBs to one on chip bus master. If two FPBs are assigned then adjacent FPBs must be assigned, for example:

- Local CPU PMBI on chip bus master: FPB0 assigned by default and software may additionally assign FPB1 for optimal local CPU performance.
- Other on chip bus masters: software may assign FPB1 and FPB2, FPB2 and FPB3, or FPB3 and FPB4.

## 6.6.3 Erase Counter and Register Accesses

The PFI to CPU point-to-point connection may be used to read the erase counter and PFRWB registers.

The access size must be SDTD else the access will error.

### **6.6.3.1 Erase Counter**

If an erase counter is accessed from a system address then  $\text{addr}[27] = 1_B$  and  $\text{addr}[19] = 0_B$ .

### **6.6.3.2 User Registers**

If a user register is accessed from a system address then  $\text{addr}[27] = 1_B$ ,  $\text{addr}[19] = 1_B$  and  $\text{addr}[18] = 0_B$ .

## 6.6.4 Safety Measures

### 6.6.4.1 Access Enable

A CPU enables read access to its Local PFLASH Bank (LPB). Access by individual on chip bus masters is allowed by enabling the appropriate TAG ID bit in the CPU LPB\_SPROT\_ACCEN\_R register.

### 6.6.4.2 ECC encoding of read data to CPU

SRI data phase ECC is encoded on the PFLASH read data sent on the PFI to CPU point-to-point connection. This ECC is decoded in the CPU for a local PFLASH access, and by the requested master for an access via SRI.

### 6.6.4.3 ECC error detection of wait cycle configuration from DMU

ECC is encoded by the DMU on the PFLASH read wait cycle configuration (recorded in DMU\_HF\_PWAIT register and transmitted by DMU to all PFI instances) and decoded within the PFI. An error is reported to the local CPU which flags the PFlash read monitor alarm to SMU. The ECC used provides 1-bit and 2-bit error detection.

### 6.6.4.4 PFI Partial Lockstep (PPL)

Control logic in PFI is lockstepped in a manner similar to CPU lockstep mechanism. The lockstepped logic (master) and the shadow logic (checker) is temporarily separated by two clock cycles. A generic comparator module is used that checks the master and shadow logic and reports an error in case of mismatch. The lockstep error is reported to the local CPU, which flags the PFLASH read path monitor alarm.

Error injection to PPL is possible through the SCU\_LCLTEST register. A failure can be injected by writing  $1_B$  to the PLCLTx bitfield in SCU\_LCLTEST register. The failure will be injected for a single cycle of the SPB clock.

The PPL also has a continuously running background self test of the lockstep comparator. For more details on the self test mechanism, please refer CPU chapter (“Lockstep Comparator Logic”).

### 6.6.4.5 Busy checker

PFI performs a check to determine if there is an operation ongoing in its associated PFLASH that was not expected by DMU. For this purpose, it checks if FSI is performing an operation on its PFLASH when the corresponding DMU\_HF\_STATUS.PxBUSY flag is not set. If the check fails, an error is generated to the CPU, resulting in PFLASH read path monitor alarm from CPU to SMU.

## 6.6.5 Revision History

**Table 210 Revision History**

Reference	Change to Previous Version	Comment
<b>V2.0.0</b>		
<b>Chapter 6.</b> 6.5	Revision History layout updated	
	Minor textual corrections. Version updated to sync with DMU and NVM Specifications and for M4 release.	
<b>V2.0.1</b>		
<b>Chapter 6.</b> 6.5	Revision History layout updated	
<b>Chapter 6.</b> 6	First-level heading structure adjusted	
<b>Chapter 6.</b> 6.4	Minor textual edits for better readability and clarity	

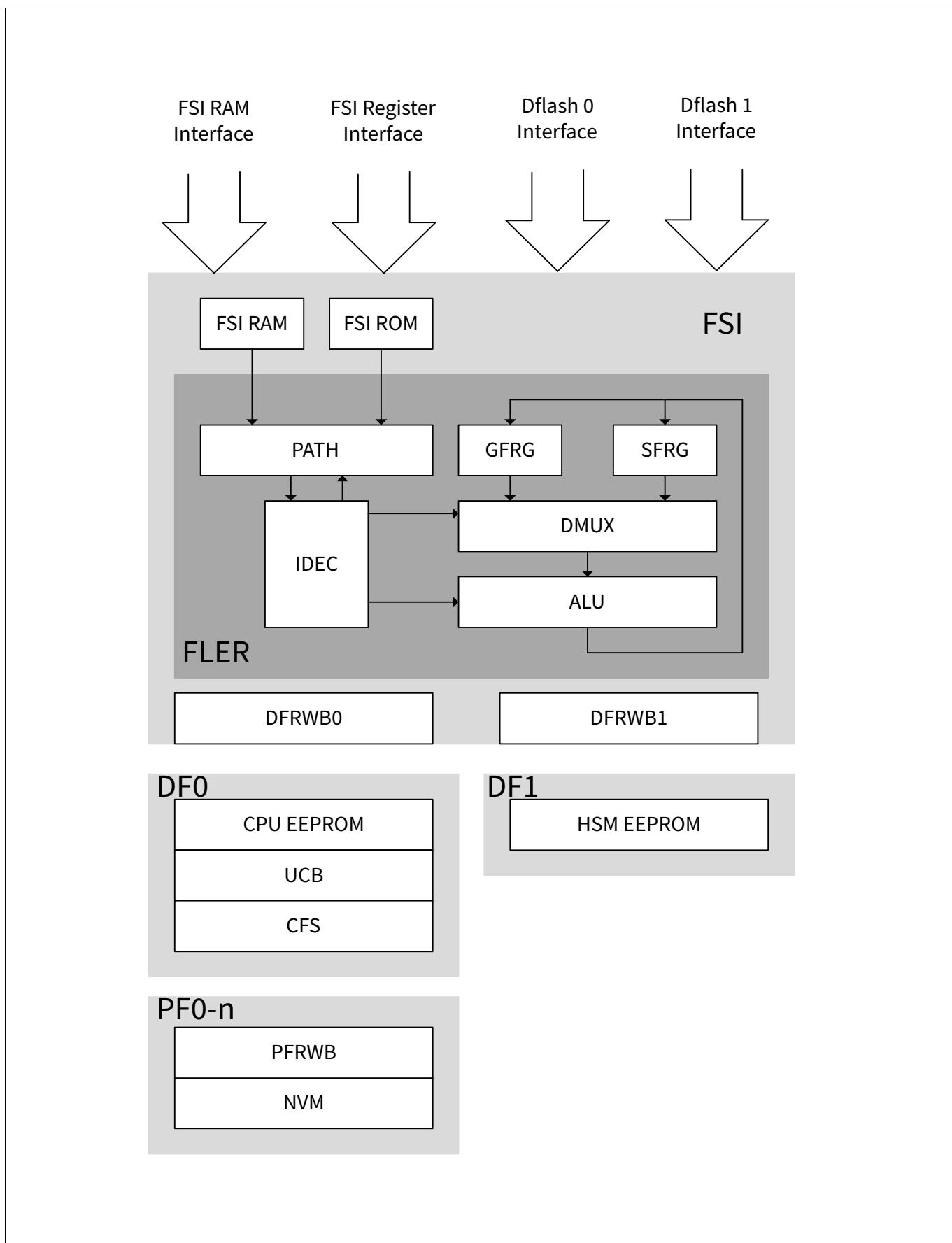
## 6.7 Non Volatile Memory (NVM)

### 6.7.1 Overview

The Non Volatile Memory component of the NVM Subsystem comprises the Program Flash (PFLASH), Data Flash (DFLASH) (including DFlash0, DFlash1, User Configuration Blocks (UCB) and Configuration Sector (CFS)), PFlash Read Write Buffer (PFRWB) and the Flash Standard Interface (FSI) modules.

- Program Flash (PFLASH): Stores program code and data constants. It is divided into one or more PFLASH banks depending on the configuration of the device. In addition to the Flash Arrays, it also contains the Analog Block with pumps and regulators. Each PFLASH has an associated PFlash Read Write Buffer (PFRWB) which provides the read data to the PFI.
- Program Flash Read Write Buffer (PFRWB): The PFRWB provides read access on the corresponding PFlash bank, status registers and Erase Counter. It performs ECC error detection, correction and flags safety alarms to SMU. PFRWB provides both uncorrected and corrected data to the PFI, which determines which one to use based on the type of access (more details in the PFI chapter).
- Data Flash (DFLASH): Utilized by user and security applications to store data. Interfaces with DFlash Read Write Buffer (DFRWB) in the FSI to provide DFlash read data. In addition to the Flash Arrays, there is an Analog Block containing pumps, regulators, reference.
- Flash Standard Interface (FSI): executes erase, program and verify operations on all flash memories.

Read accesses to the NVM are memory mapped reads. NVM programming and erase operations are sequenced by the Flash Standard Interface (FSI) micro controller.



**Figure 65 Block Diagram of the NVM module.**

## 6.7.2 Functional Description of the Flash Standard Interface (FSI)

The Flash Standard Interface (FSI) (in [Figure 65](#)) manages all maintenance tasks (start up, erasing, programming, verifying, analysis, etc.) of the PFLASH and DFLASH memories.

The FSI consists of the following hardware parts:

- FSI RAM
  - Shared SRAM for program code and Assembly Buffer (ASB) write data.
  - 64-bit read/write interface
- FSI ROM
  - Bootstrap program code.
  - Stable command sequences.
- Flash Processor (FLER)
  - 8-bit pipelined RISC processor
  - 8-bit read/write interface to Core Function Registers (CFR) and Special Function Registers (SFR)

### 6.7.2.1 FSI ROM

The FSI ROM stores bootstrap program code and stable firmware routines.

### 6.7.2.2 FSI SFR

The FSI is controlled by accesses to the FSI Special Function Registers.

#### 6.7.2.2.1 FSI SFR Access Control

FSI SFR access control is as follows:

##### FSI SFR Access Size

Only byte wide accesses are supported to the 8-bit FSI SFRs. Other access sizes generate a bus error.

Byte-size only access is fully supported by TriCore compilers. A register defined as volatile unsigned char is only ever accessed with byte-size load or byte-size store operations.

##### FSI SFR Accesses in Operation Mode

The following SFRs may be accessed via the DMU SRI slave interface:

- FSI Host communication SFRs
  - [COMM\\_1](#)
  - [COMM\\_2](#)
- FSI HSM communication SFRs
  - [HSMCOMM\\_1](#)
  - [HSMCOMM\\_2](#)

If the DMU is not in a Flash Test Mode then accesses to all other FSI SFRs generate a bus error.

##### FSI SFR Accesses in Sleep Mode

All accesses fail and generate a bus error.

*Note: Accesses after requesting Sleep Mode but before Sleep Mode is reached may result in erroneous behavior.*

### 6.7.2.3 Communication with FSI

The Host communicates with FSI via the Host Command Sequence Interpreter (Host CSI).

The HSM communicates with FSI via the HSM Command Sequence Interpreter (HSM CSI).

The HSM CSI interprets writes to the DF1 address range. DF1 access is controlled via DMU\_SP\_PROCONHSMCFG.HSMDX (at [Table 211](#)):

**Table 211 DF1 Access Control**

HSMDX	DF1 Access
0 <sub>B</sub>	DF1 is accessed via the Host CSI by all masters and via the HSM CSI by HSM master.
1 <sub>B</sub>	DF1 is accessed via the HSM CSI by HSM master only.

Control and status information is passed via the FSI communication SFRs:

- Host CSI: [COMM\\_1](#) and [COMM\\_2](#)
- HSM CSI: [HSMCOMM\\_1](#) and [HSMCOMM\\_2](#)

### 6.7.2.3.1 DMU Command Sequences

If a command is initiated by a DMU command sequence in Operation Mode then access to the FSI communication SFRs is blocked as follows:

- [COMM\\_1](#) and [COMM\\_2](#) accesses are blocked when any PFLASH or DFLASH BUSY reports a flash busy flag in the DMU\_HF\_STATUS register.
- [HSMCOMM\\_1](#) and [HSMCOMM\\_2](#) accesses are blocked when DF1 BUSY reports a flash busy flag in the DMU\_SF\_STATUS register.

## 6.7.3 Registers

### 6.7.3.1 FSI Registers

The FSI Registers are accessed through the DMU SRI slave interface. The register set includes access protection registers and NVM configuration, control and status registers.

#### Register Accesses

Only byte wide accesses are supported to the FSI Registers. Other access sizes generate a bus error on the DMU SRI slave interface.

**Table 212 Register Address Space - FSI**

Module	Base Address	End Address	Note
FSI	F8030000 <sub>H</sub>	F80300FF <sub>H</sub>	sri slave interface

**Table 213 Register Overview - FSI (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
COMM_1	Communication Register 1	0004 <sub>H</sub>	U,SV	U,SV	System Reset	<a href="#">171</a>
COMM_2	Communication Register 2	0005 <sub>H</sub>	U,SV	U,SV	System Reset	<a href="#">172</a>
HSMCOMM_1	HSM Communication Register 1	0006 <sub>H</sub>	H	H	System Reset	<a href="#">172</a>
HSMCOMM_2	HSM Communication Register 2	0007 <sub>H</sub>	H	H	System Reset	<a href="#">172</a>

#### 6.7.3.1.1 Status register

##### Communication Register 1

COMM_1							
Communication Register 1				System Reset Value: 00 <sub>H</sub>			
7	6	5	4	3	2	1	0
COMM1							
rw							

Field	Bits	Type	Description
COMM1	7:0	rw	<b>FSI Communication 1</b> This register can be written by FSI and DMU and is used to give status/handshake information between FSI and DMU.

## Communication Register 2

### COMM\_2

#### Communication Register 2

(0005<sub>H</sub>)

System Reset Value: 00<sub>H</sub>

7	6	5	4	3	2	1	0
COMM2							
rw							

Field	Bits	Type	Description
COMM2	7:0	rw	<b>FSI Communication 2</b> This register can be written by FSI and DMU and is used to give status/handshake information between FSI and DMU.

## HSM Communication Register 1

### HSMCOMM\_1

#### HSM Communication Register 1

(0006<sub>H</sub>)

System Reset Value: 00<sub>H</sub>

7	6	5	4	3	2	1	0
HSMCOMM1							
rw							

Field	Bits	Type	Description
HSMCOMM1	7:0	rw	<b>HSM FSI Communication 1</b> This register can be written by FSI and DMU and is used to give status/handshake information between FSI and DMU.

## HSM Communication Register 2

### HSMCOMM\_2

#### HSM Communication Register 2

(0007<sub>H</sub>)

System Reset Value: 00<sub>H</sub>

7	6	5	4	3	2	1	0
HSMCOMM2							
rw							

Field	Bits	Type	Description
HSMCOMM2	7:0	rw	<b>HSM FSI Communication 2</b> This register can be written by FSI and DMU and is used to give status/handshake information between FSI and DMU.

### 6.7.3.2 PFRWB (PFI) Registers

The PFRWB User and Test Registers are accessed through the PFI. The local CPU can read the registers on the direct PFI to CPU point to point connection. A remote CPU can read these registers by triggering a request on the SRI SIF of the local CPU.

#### Access Protection

The PFRWB User and Test Registers are read-hardware.

**Table 214 Register Address Space - PFI**

Module	Base Address	End Address	Note
(PFI0)	80000000 <sub>H</sub>	802FFFFF <sub>H</sub>	Program Flash cached address space
	A0000000 <sub>H</sub>	A02FFFFF <sub>H</sub>	Program Flash non-cached address space
	A8000000 <sub>H</sub>	A8003FFF <sub>H</sub>	Erase Counter address space
PFI0	A8080000 <sub>H</sub>	A80FFFFF <sub>H</sub>	Register address space
(PFI1)	80300000 <sub>H</sub>	805FFFFF <sub>H</sub>	Program Flash cached address space
	A0300000 <sub>H</sub>	A05FFFFF <sub>H</sub>	Program Flash non-cached address space
	A8300000 <sub>H</sub>	A8303FFF <sub>H</sub>	Erase Counter address space
PFI1	A8380000 <sub>H</sub>	A83FFFFF <sub>H</sub>	Register address space
(PFI2)	80600000 <sub>H</sub>	808FFFFF <sub>H</sub>	Program Flash cached address space
	A0600000 <sub>H</sub>	A08FFFFF <sub>H</sub>	Program Flash non-cached address space
	A8600000 <sub>H</sub>	A8603FFF <sub>H</sub>	Erase Counter address space
PFI2	A8680000 <sub>H</sub>	A86FFFFF <sub>H</sub>	Register address space
(PFI3)	80900000 <sub>H</sub>	80BFFFFF <sub>H</sub>	Program Flash cached address space
	A0900000 <sub>H</sub>	A0BFFFFF <sub>H</sub>	Program Flash non-cached address space
	A8900000 <sub>H</sub>	A8903FFF <sub>H</sub>	Erase Counter address space
PFI3	A8980000 <sub>H</sub>	A89FFFFF <sub>H</sub>	Register address space
(PFI4)	80C00000 <sub>H</sub>	80EFFFFF <sub>H</sub>	Program Flash cached address space
	A0C00000 <sub>H</sub>	A0EFFFFF <sub>H</sub>	Program Flash non-cached address space
	A8C00000 <sub>H</sub>	A8C03FFF <sub>H</sub>	Erase Counter address space
PFI4	A8C80000 <sub>H</sub>	A8CFFFFF <sub>H</sub>	Register address space
(PFI5)	80F00000 <sub>H</sub>	80FFFFFF <sub>H</sub>	Program Flash cached address space
	A0F00000 <sub>H</sub>	A0FFFFFF <sub>H</sub>	Program Flash non-cached address space
	A8F00000 <sub>H</sub>	A8F03FFF <sub>H</sub>	Erase Counter address space
PFI5	A8F80000 <sub>H</sub>	A8FFFFFF <sub>H</sub>	Register address space

**Table 215 Register Overview - PFI (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
ECCR	ECC Read Register	000000 <sub>H</sub>	P,U,SV	BE	System Reset	<a href="#">174</a>
ECCS	ECC Status Register	000020 <sub>H</sub>	P,U,SV	BE	System Reset	<a href="#">174</a>

Table 215 Register Overview - PFI (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
SBABRECORDx	SBAB Record x	002000 <sub>H</sub> +x*20 <sub>H</sub>	P,U,SV	BE	System Reset	<a href="#">177</a>
DBABRECORDx	DBAB Record x	004000 <sub>H</sub> +x*20 <sub>H</sub>	P,U,SV	BE	System Reset	<a href="#">178</a>
MBABRECORDx	MBAB Record 0	008000 <sub>H</sub>	P,U,SV	BE	System Reset	<a href="#">179</a>
ZBABRECORDx	ZBAB Record x	00C000 <sub>H</sub> +x*20 <sub>H</sub>	P,U,SV	BE	System Reset	<a href="#">180</a>

**Attention:** The Bitline Redundancy Registers, Sector Redundancy Registers and Test Registers do not down configure with PFLASH product configuration.

### 6.7.3.2.1 PFI ECC Registers

#### ECC Read Register

The ECC Read Register shall store the ECC checksum read during the last PFLASH Bank NVM read access.

ECCR															
ECC Read Register (0000000 <sub>H</sub> ) System Reset Value: 0000 0000 <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES								R22		RCODE					
r								r					rh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCODE															
r															

Field	Bits	Type	Description
<b>RCODE</b>	21:0	rh	<b>Error Correction Read Code</b> ECC code, read from the Flash read buffer with last data read operation.
<b>R22</b>	23:22	r	<b>Reserved - RES</b> Reserved
<b>RES</b>	31:24	r	<b>Reserved</b> Always read as 0; should be written with 0.

#### ECC Status Register

The ECC Status Register must capture ECC errors detected during the last PFLASH Bank NVM read access.

**Note:** All status bits in this register are cleared when CPUx\_FLASHCON2.ECCSCLR is written to 01<sub>B</sub>.

**ECCS****ECC Status Register**(0000020<sub>H</sub>)**System Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								AERAN Y	AAL1	AAL0	ARRA	AERM	RES	AER2	AER1
					r			rh	rh	rh	rh	rh	r	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								ERRA NY	ALL1	ALLO	ERRA	ERRM	RES	ERR2	ERR1
					r			rh	rh	rh	rh	rh	r	rh	rh

Field	Bits	Type	Description
<b>ERR1</b>	0	rh	<b>Read Access Single Bit ECC Error</b> The flag reports a single bit ECC failure during the last NVM read access. 0 <sub>B</sub> No single bit ECC failure occurred. 1 <sub>B</sub> A single bit ECC failure occurred.
<b>ERR2</b>	1	rh	<b>Read Access Double Bit ECC Error</b> The flag reports a double bit ECC failure during the last NVM read access. 0 <sub>B</sub> No double bit ECC failure occurred. 1 <sub>B</sub> A double bit ECC failure occurred.
<b>RES</b>	2, 15:8, 18, 31:24	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>ERRM</b>	3	rh	<b>Read Access Multi-bit ECC Error</b> The flag reports multi bit ECC failure during the last NVM read access. 0 <sub>B</sub> No multi bit ECC failure occurred. 1 <sub>B</sub> Multi bit ECC failure occurred.
<b>ERRA</b>	4	rh	<b>Read Access ECC Error Within the Address</b> The flag reports an address error during the last NVM read access. 0 <sub>B</sub> No Address error detected. 1 <sub>B</sub> Address detected.
<b>ALLO</b>	5	rh	<b>Read Access All Zeros</b> The flag reports the All Zeros condition during the last NVM read access. 0 <sub>B</sub> No All Zeros detected. 1 <sub>B</sub> All zeros detected.
<b>ALL1</b>	6	rh	<b>All Ones</b> The flag reports the All Ones condition during the last NVM read access. 0 <sub>B</sub> No All Ones detected. 1 <sub>B</sub> All ones detected.
<b>ERRANY</b>	7	rh	<b>Any Read Access ECC Error</b> The flag reports any ECC failure during the last NVM read access. 0 <sub>B</sub> No ECC failure occurred. 1 <sub>B</sub> ECC failure occurred.

Field	Bits	Type	Description
<b>AER1</b>	16	rh	<b>Accumulated Single Bit ECC Errors</b> The flag accumulates single bit failures during NVM read operations. $0_B$ No single bit ECC failure occurred. $1_B$ At least one single bit ECC failure occurred.
<b>AER2</b>	17	rh	<b>Accumulated Double Bit ECC Errors</b> The flag accumulates double bit failures during NVM read operations. $0_B$ No double bit ECC failure occurred. $1_B$ At least one double bit ECC failure occurred.
<b>AERM</b>	19	rh	<b>Accumulated Multi-bit ECC Errors</b> The flag accumulates multi bit failures during NVM read accesses. $0_B$ No multi bit ECC failure occurred. $1_B$ Multi bit ECC failure occurred.
<b>ARRA</b>	20	rh	<b>Accumulated ECC Error Within the Address</b> The flag accumulates an address errors during NVM read accesses. $0_B$ No Address error detected. $1_B$ Address detected.
<b>AAL0</b>	21	rh	<b>Accumulated All Zeros</b> The flag accumulates the All Zeros condition during NVM read accesses. $0_B$ No All Zeros detected. $1_B$ All zeros detected.
<b>AAL1</b>	22	rh	<b>Accumulated All Ones</b> The flag accumulates the All Ones condition during NVM read accesses. $0_B$ No All Ones detected. $1_B$ All ones detected.
<b>AERANY</b>	23	rh	<b>Accumulated Any Read Access ECC Error</b> The status bit accumulates ECC failures during NVM read accesses. $0_B$ No ECC failure occurred. $1_B$ ECC failure occurred.

Note: An address error (ERRA) is also flagged as a multi bit error (ERRM). Only 1 or 2 bit address errors can be clearly identified as address errors. Greater than 2 bit address errors appear as multi bit error and might not be flagged as address error.

### 6.7.3.2.2 PFI Corrected Single Bits Address Buffer (SBAB)

When data is read from Flash NVM and the ECC decoder detects a correctable single bit error, then the local address is stored in the SBAB. Each local address is only entered once, and covers 256 bits of data. The bottom five reserved bits of the SBAB read as 0, and can be concatenated with the local address to give the local address as seen by the system (without the base offset).

When the SBAB becomes full the SMU is informed which can trigger an interrupt and the DBAB limit is changed to 1.

#### SBAB Record x

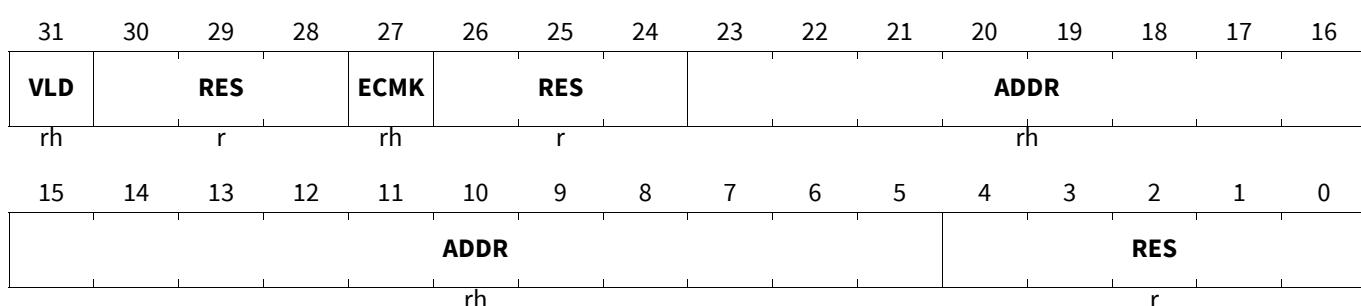
*Note:* All bits in this register are cleared when CPUx\_FLASHCON2.SBABCLR is written to 01<sub>B</sub>.

The address of the data with a detected error is stored in a SBAB Recorder Register.

Reset: system reset.

#### SBABRECORDx (x=0-16)

**SBAB Record x** (002000<sub>H</sub>+x\*20<sub>H</sub>) System Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>RES</b>	4:0, 26:24, 30:28	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>ADDR</b>	23:5	rh	<b>Address</b> Captured local PFLASH bank address of the page with detected error.
<b>ECMK</b>	27	rh	<b>Erase Counter Marker</b> The captured address is from the Flash or Erase Counter. 0 <sub>B</sub> Erroneous page is in the PFLASH bank. 1 <sub>B</sub> Erroneous page is in the erase counter.
<b>VLD</b>	31	rh	<b>Valid</b> 0 <sub>B</sub> No entry recorded. 1 <sub>B</sub> Valid entry recorded.

### 6.7.3.2.3 PFI Corrected Double Bits Address Buffer (DBAB)

When data is read from Flash NVM and the ECC decoder detects a correctable double bit error, then the local address is stored in the DBAB. Each local address is only entered once, and covers 256 bits of data. The bottom five reserved bits of the DBAB read as 0, and can be concatenated with the local address to give the local address as seen by the system (without the base offset).

An alarm to SMU is generated when DBAB becomes full, which can trigger an interrupt.

The maximum number of DBAB records will change from two to one when SBAB becomes full.

#### DBAB Record x

Note: All bits in this register are cleared when  $CPUx\_FLASHCON2.DBABCLR$  is written to  $01_B$ .

The address of the data with a detected error is stored in a DBAB Recorder Register.

Reset: system reset.

#### DBABRECORD $x$ ( $x=0-1$ )

<b>DBAB Record x</b>															<b>(004000<sub>H</sub>+x*20<sub>H</sub>)</b>	<b>System Reset Value: 0000 0000<sub>H</sub></b>		
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16			
<b>VLD</b>	<b>RES</b>			<b>ECMK</b>	<b>RES</b>						<b>ADDR</b>							
rh	r			rh	r						rh							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
<b>ADDR</b>													<b>RES</b>					
rh													r					

Field	Bits	Type	Description
<b>RES</b>	4:0, 26:24, 30:28	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>ADDR</b>	23:5	rh	<b>Address</b> Captured address of local PFLASH bank with detected error.
<b>ECMK</b>	27	rh	<b>Erase Counter Marker</b> The captured address is from the Flash or Erase Counter. $0_B$ Erroneous page is in the PFLASH bank. $1_B$ Erroneous page is in the erase counter.
<b>VLD</b>	31	rh	<b>Valid</b> $0_B$ No entry recorded. $1_B$ Valid entry recorded.

### 6.7.3.2.4 PFI Uncorrected Multi Bits Address Buffer (MBAB)

When data is read from Flash NVM and the ECC decoder detects an uncorrected multi bit error (including all-0 and all-1 errors) then the local address is stored in the MBAB. Each local address is only entered once, and covers 256 bits of data. The bottom five reserved bits of the MBAB read as 0, and can be concatenated with the local address to give the local address as seen by the system (without the base offset).

When the MBAB becomes full the SMU is informed which can trigger an interrupt.

#### MBAB Record 0

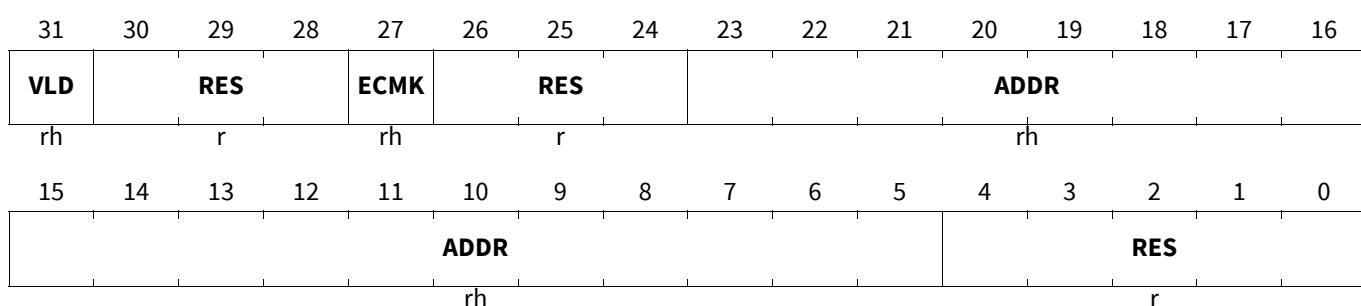
Note: *All bits in this register are cleared when CPUx\_FLASHCON2.MBABCLR is written to 01<sub>B</sub>.*

The address of the data with a detected error is stored in a MBAB Recorder Register.

Reset: system reset.

#### MBABRECORD0

**MBAB Record 0** **(008000<sub>H</sub>)** **System Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>RES</b>	4:0, 26:24, 30:28	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>ADDR</b>	23:5	rh	<b>Address</b> Captured address of local PFLASH bank with detected error.
<b>ECMK</b>	27	rh	<b>Erase Counter Marker</b> The captured address is from the Flash or Erase Counter. 0 <sub>B</sub> Erroneous page is in the PFLASH bank. 1 <sub>B</sub> Erroneous page is in the erase counter.
<b>VLD</b>	31	rh	<b>Valid</b> 0 <sub>B</sub> No entry recorded. 1 <sub>B</sub> Valid entry recorded.

### 6.7.3.2.5 PFI Uncorrected All Zeros Bits Address Buffer (ZBAB)

When data is read from Flash NVM and the ECC decoder detects an all-0 error, then the local address at the PFRWB is stored in the ZBAB. Each local address is only entered once, and covers 256 bits of data. The bottom five reserved bits of the ZBAB read as 0, and can be concatenated with the local address to give the local address as seen by the system (without the base offset).

When the ZBAB becomes full the SMU is informed which can trigger an interrupt.

#### ZBAB Record x

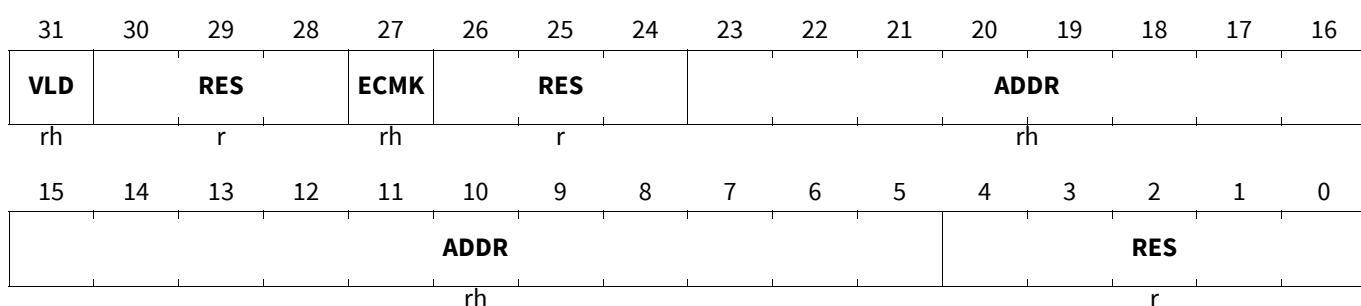
Note: *All bits in this register are cleared when CPUx\_FLASHCON2.ZBABCLR is written to 01<sub>B</sub>.*

The address of the data with a detected error is stored in a ZBAB Recorder Register.

Reset: system reset.

#### ZBABRECORDx (x=0-3)

**ZBAB Record x** (00C000<sub>H</sub>+x\*20<sub>H</sub>) System Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>RES</b>	4:0, 26:24, 30:28	r	<b>Reserved</b> Always read as 0; should be written with 0.
<b>ADDR</b>	23:5	rh	<b>Address</b> Captured address of local PFLASH bank with detected error.
<b>ECMK</b>	27	rh	<b>Erase Counter Marker</b> The captured address is from the Flash or Erase Counter. 0 <sub>B</sub> Erroneous page is in the PFLASH bank. 1 <sub>B</sub> Erroneous page is in the erase counter.
<b>VLD</b>	31	rh	<b>Valid</b> 0 <sub>B</sub> No entry recorded. 1 <sub>B</sub> Valid entry recorded.

## 6.7.4 Revision History

**Table 216 Revision History**

Reference	Change to Previous Version	Comment
<b>V2.0.4</b>		
<a href="#">Page 180</a>	<a href="#">Chapter 6.7.3.2.5</a> - Improved description of address stored and meaning of reserved bits 4:0.	
<a href="#">Page 179</a>	<a href="#">Chapter 6.7.3.2.4</a> - Improved description of address stored and meaning of reserved bits 4:0.	
<a href="#">Page 178</a>	<a href="#">Chapter 6.7.3.2.3</a> - Improved description of address stored and meaning of reserved bits 4:0.	
<a href="#">Page 177</a>	<a href="#">Chapter 6.7.3.2.2</a> - Improved description of address stored and meaning of reserved bits 4:0.	
<a href="#">Page 180</a>	<a href="#">Chapter 6.7.3.2.5</a> - Added ECMK field to the register with description.	
<a href="#">Page 179</a>	<a href="#">Chapter 6.7.3.2.4</a> - Added ECMK field to the register with description.	
<a href="#">Page 178</a>	<a href="#">Chapter 6.7.3.2.3</a> - Added ECMK field to the register with description.	
<a href="#">Page 177</a>	<a href="#">Chapter 6.7.3.2.2</a> - Added ECMK field to the register with description.	
<a href="#">Page 180</a>	<a href="#">Chapter 6.7.3.2.5</a> - Clarified what is cleared by FLASHCON2.ZBABCLR write.	
<a href="#">Page 179</a>	<a href="#">Chapter 6.7.3.2.4</a> - Clarified what is cleared by FLASHCON2.MBABCLR write.	
<a href="#">Page 178</a>	<a href="#">Chapter 6.7.3.2.3</a> - Clarified what is cleared by FLASHCON2.DBABCLR write.	
<a href="#">Page 177</a>	<a href="#">Chapter 6.7.3.2.2</a> - Clarified what is cleared by FLASHCON2.SBABCLR write.	
<b>V2.0.5</b>		
<a href="#">Page 174</a>	<a href="#">ECC Status Register</a> - Corrected description of value 0 for AAL0 and AAL1, missing negation..	
<b>V2.0.6</b>		
<a href="#">Page 170</a>	<a href="#">Chapter 6.7.2.3</a> - Changed DMU_SF_PROCONHSM to DMU_SP_PROCONHSMCFG .	

## 6.8 User Configuration Block (UCB)

This chapter describes the content of the User Configuration Block (not the implementation of this storage area in Flash).

### 6.8.1 Overview

The User Configuration Block contains information used for configuration and protection installation. It is part of Data Flash 0.

### 6.8.2 UCB Address Map

All UCB offset address are given with reference to the following “Module” base address.

**Table 217 Register Address Space - UCB**

Module	Base Address	End Address	Note
UCB	AF400000 <sub>H</sub>	AF405FFF <sub>H</sub>	UCB Range

#### 6.8.2.1 List of Defined UCBs

The following overview table shows a list of all defined UCBs:

**Table 218 On Chip Bus Address Map of User Configuration Blocks**

Address Range	Size	Unit	Access Type	
			Read	Write
AF40 0000 <sub>H</sub> - AF40 01FF <sub>H</sub>	512 Byte	UCB00 (UCB_BMHD0_ORIG)	Access	SRIBE
AF40 0200 <sub>H</sub> - AF40 03FF <sub>H</sub>	512 Byte	UCB01 (UCB_BMHD1_ORIG)	Access	SRIBE
AF40 0400 <sub>H</sub> - AF40 05FF <sub>H</sub>	512 Byte	UCB02 (UCB_BMHD2_ORIG)	Access	SRIBE
AF40 0600 <sub>H</sub> - AF40 07FF <sub>H</sub>	512 Byte	UCB03 (UCB_BMHD3_ORIG)	Access	SRIBE
AF40 0800 <sub>H</sub> - AF40 09FF <sub>H</sub>	512 Byte	UCB04 (UCB_SSW)	Access	SRIBE
AF40 0A00 <sub>H</sub> - AF40 0BFF <sub>H</sub>	512 Byte	UCB05 (UCB_USER)	Access	SRIBE
AF40 0C00 <sub>H</sub> - AF40 0DFF <sub>H</sub>	512 Byte	UCB06 (UCB_TEST)	Access	SRIBE
AF40 0E00 <sub>H</sub> - AF40 0FFF <sub>H</sub>	512 Byte	UCB07 (UCB_HSMCFG)	Access	SRIBE
AF40 1000 <sub>H</sub> - AF40 11FF <sub>H</sub>	512 Byte	UCB08 (UCB_BMHD0_COPY)	Access	SRIBE
AF40 1200 <sub>H</sub> - AF40 13FF <sub>H</sub>	512 Byte	UCB09 (UCB_BMHD1_COPY)	Access	SRIBE
AF40 1400 <sub>H</sub> - AF40 15FF <sub>H</sub>	512 Byte	UCB10 (UCB_BMHD2_COPY)	Access	SRIBE
AF40 1600 <sub>H</sub> - AF40 17FF <sub>H</sub>	512 Byte	UCB11 (UCB_BMHD3_COPY)	Access	SRIBE
AF40 1800 <sub>H</sub> - AF40 19FF <sub>H</sub>	512 Byte	UCB12 (UCB_REDSEC)	Access	SRIBE
AF40 1A00 <sub>H</sub> - AF40 1BFF <sub>H</sub>	512 Byte	UCB13 (Reserved)	Access	SRIBE
AF40 1C00 <sub>H</sub> - AF40 1DFF <sub>H</sub>	512 Byte	UCB14 (Reserved)	Access	SRIBE
AF40 1E00 <sub>H</sub> - AF40 1FFF <sub>H</sub>	512 Byte	UCB15 (UCB_RETEST)	Access	SRIBE
AF40 2000 <sub>H</sub> - AF40 21FF <sub>H</sub>	512 Byte	UCB16 (UCB_PFLASH_ORIG)	Access	SRIBE
AF40 2200 <sub>H</sub> - AF40 23FF <sub>H</sub>	512 Byte	UCB17 (UCB_DFLASH_ORIG)	Access	SRIBE
AF40 2400 <sub>H</sub> - AF40 25FF <sub>H</sub>	512 Byte	UCB18 (UCB_DBG_ORIG)	Access	SRIBE
AF40 2600 <sub>H</sub> - AF40 27FF <sub>H</sub>	512 Byte	UCB19 (UCB_HSM_ORIG)	Access	SRIBE

**Table 218 On Chip Bus Address Map of User Configuration Blocks (cont'd)**

Address Range	Size	Unit	Access Type	
			Read	Write
AF40 2800 <sub>H</sub> - AF40 29FF <sub>H</sub>	512 Byte	UCB20 (UCB_HSMCOTP0_ORIG)	Access	SRIBE
AF40 2A00 <sub>H</sub> - AF40 2BFF <sub>H</sub>	512 Byte	UCB21 (UCB_HSMCOTP1_ORIG)	Access	SRIBE
AF40 2C00 <sub>H</sub> - AF40 2DFF <sub>H</sub>	512 Byte	UCB22 (UCB_ECPPIO_ORIG)	Access	SRIBE
AF40 2E00 <sub>H</sub> - AF40 2FFF <sub>H</sub>	512 Byte	UCB23 (UCB_SWAP_ORIG)	Access	SRIBE
AF40 3000 <sub>H</sub> - AF40 31FF <sub>H</sub>	512 Byte	UCB24 (UCB_PFLASH_COPY)	Access	SRIBE
AF40 3200 <sub>H</sub> - AF40 33FF <sub>H</sub>	512 Byte	UCB25 (UCB_DFLASH_COPY)	Access	SRIBE
AF40 3400 <sub>H</sub> - AF40 35FF <sub>H</sub>	512 Byte	UCB26 (UCB_DBG_COPY)	Access	SRIBE
AF40 3600 <sub>H</sub> - AF40 37FF <sub>H</sub>	512 Byte	UCB27 (UCB_HSM_COPY)	Access	SRIBE
AF40 3800 <sub>H</sub> - AF40 39FF <sub>H</sub>	512 Byte	UCB28 (UCB_HSMCOTP0_COPY)	Access	SRIBE
AF40 3A00 <sub>H</sub> - AF40 3BFF <sub>H</sub>	512 Byte	UCB29 (UCB_HSMCOTP1_COPY)	Access	SRIBE
AF40 3C00 <sub>H</sub> - AF40 3DFF <sub>H</sub>	512 Byte	UCB30 (UCB_ECPPIO_COPY)	Access	SRIBE
AF40 3E00 <sub>H</sub> - AF40 3FFF <sub>H</sub>	512 Byte	UCB31 (UCB_SWAP_COPY)	Access	SRIBE
AF40 4000 <sub>H</sub> - AF40 41FF <sub>H</sub>	512 Byte	UCB32 (UCB OTP0_ORIG)	Access	SRIBE
AF40 4200 <sub>H</sub> - AF40 43FF <sub>H</sub>	512 Byte	UCB33 (UCB OTP1_ORIG)	Access	SRIBE
AF40 4400 <sub>H</sub> - AF40 45FF <sub>H</sub>	512 Byte	UCB34 (UCB OTP2_ORIG)	Access	SRIBE
AF40 4600 <sub>H</sub> - AF40 47FF <sub>H</sub>	512 Byte	UCB35 (UCB OTP3_ORIG)	Access	SRIBE
AF40 4800 <sub>H</sub> - AF40 49FF <sub>H</sub>	512 Byte	UCB36 (UCB OTP4_ORIG)	Access	SRIBE
AF40 4A00 <sub>H</sub> - AF40 4BFF <sub>H</sub>	512 Byte	UCB37 (UCB OTP5_ORIG)	Access	SRIBE
AF40 4C00 <sub>H</sub> - AF40 4DFF <sub>H</sub>	512 Byte	UCB38 (UCB OTP6_ORIG)	Access	SRIBE
AF40 4E00 <sub>H</sub> - AF40 4FFF <sub>H</sub>	512 Byte	UCB39 (UCB OTP7_ORIG)	Access	SRIBE
AF40 5000 <sub>H</sub> - AF40 51FF <sub>H</sub>	512 Byte	UCB40 (UCB OTP0_COPY)	Access	SRIBE
AF40 5200 <sub>H</sub> - AF40 53FF <sub>H</sub>	512 Byte	UCB41 (UCB OTP1_COPY)	Access	SRIBE
AF40 5400 <sub>H</sub> - AF40 55FF <sub>H</sub>	512 Byte	UCB42 (UCB OTP2_COPY)	Access	SRIBE
AF40 5600 <sub>H</sub> - AF40 57FF <sub>H</sub>	512 Byte	UCB43 (UCB OTP3_COPY)	Access	SRIBE
AF40 5800 <sub>H</sub> - AF40 59FF <sub>H</sub>	512 Byte	UCB44 (UCB OTP4_COPY)	Access	SRIBE
AF40 5A00 <sub>H</sub> - AF40 5BFF <sub>H</sub>	512 Byte	UCB45 (UCB OTP5_COPY)	Access	SRIBE
AF40 5C00 <sub>H</sub> - AF40 5DFF <sub>H</sub>	512 Byte	UCB46 (UCB OTP6_COPY)	Access	SRIBE
AF40 5E00 <sub>H</sub> - AF40 5FFF <sub>H</sub>	512 Byte	UCB47 (UCB OTP7_COPY)	Access	SRIBE

The following detailed UCB tables contain a reference page number if the corresponding entries needs more explanation. Entries without page number are self explaining.

### 6.8.2.2 UCB\_BMHDx\_ORIG and UCB\_BMHDx\_COPY (x = 0 - 3)

The four UCB\_BMHDx\_ORIG (UCB00, UCB01, UCB02, UCB03) and UCB\_BMHDx\_COPY (UCB08, UCB09, UCB10, UCB11) are used by the customer to configure the Boot Mode Headers (BMHD). These are evaluated by the SSW.

In the following only the detailed offset tables for UCB00 and UCB01 are shown. UCB00 serves as example for a UCB\_BMHD with password (also UCB08) and UCB01 as example for UCB\_BMHD without password (also UCB02, UCB03, UCB09, UCB10, UCB11) are shown. The other UCBs have the same content just based on different offset addresses.

## Delivery State

The “CONFIRMATION” code entries contain the UNLOCKED value (see “DMU” chapter, section “UCB Confirmation”). All other addresses are delivered erased.

**Table 219 Register Overview - UCB00 (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
BMI_BMH DID	UCB_BMHD0_ORIG_DATA - Boot Mode Index (BMI) and Boot Mode Header ID (CODE) = B359H	0000 <sub>H</sub>	
STAD	UCB_BMHD0_ORIG_DATA - ABHMDx start address (in case BMI.HWCFG = ABM = 110B) or User Code start address (in case BMI.HWCFG = Flash start = 111B)	0004 <sub>H</sub>	
CRCBMHD	UCB_BMHD0_ORIG_DATA - Check Result for the BMI Header (offset 000H - 007H)	0008 <sub>H</sub>	
CRCBMHD_N	UCB_BMHD0_ORIG_DATA - Inverted Check Result for the BMI Header (offset 000H - 007H)	000C <sub>H</sub>	
PW0	UCB_BMHD0_ORIG_PW - 256-bit password protection, PW0 (least significant)	0100 <sub>H</sub>	
PW1	UCB_BMHD0_ORIG_PW - 256-bit password protection, PW1	0104 <sub>H</sub>	
PW2	UCB_BMHD0_ORIG_PW - 256-bit password protection, PW2	0108 <sub>H</sub>	
PW3	UCB_BMHD0_ORIG_PW - 256-bit password protection, PW3	010C <sub>H</sub>	
PW4	UCB_BMHD0_ORIG_PW - 256-bit password protection, PW4	0110 <sub>H</sub>	
PW5	UCB_BMHD0_ORIG_PW - 256-bit password protection, PW5	0114 <sub>H</sub>	
PW6	UCB_BMHD0_ORIG_PW - 256-bit password protection, PW6	0118 <sub>H</sub>	
PW7	UCB_BMHD0_ORIG_PW - 256-bit password protection, PW7	011C <sub>H</sub>	
CONFIRMATION	UCB_BMHD0_ORIG_CODE - 32-bit CODE	01F0 <sub>H</sub>	

**Table 220 Register Overview - UCB01 (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
BMI_BMH DID	UCB_BMHD1_ORIG_DATA - Boot Mode Index (BMI) and Boot Mode Header ID (CODE) = B359H	0200 <sub>H</sub>	
STAD	UCB_BMHD1_ORIG_DATA - ABHMDx start address (in case BMI.HWCFG = ABM = 110B) or User Code start address (in case BMI.HWCFG = Flash start = 111B)	0204 <sub>H</sub>	
CRCBMHD	UCB_BMHD1_ORIG_DATA - Check Result for the BMI Header (offset 000H - 007H)	0208 <sub>H</sub>	
CRCBMHD_N	UCB_BMHD1_ORIG_DATA - Inverted Check Result for the BMI Header (offset 000H - 007H)	020C <sub>H</sub>	
CONFIRMATION	UCB_BMHD1_ORIG_CODE - 32-bit CODE	03F0 <sub>H</sub>	

### 6.8.2.3 UCB\_SSW

The UCB\_SSW contains data supplied by Infineon and predominantly used by the SSW to configure the device. The single entries are not published with exception of the LBIST configuration and expected signature entries SCU\_LBISTCTRL\*. These can be also used by application SW when executing the LBIST.

**Table 221 Register Overview - UCB04 (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
SCU_LBISTCTRL3_BODY0	SCU_LBISTCTRL3_BODY0 - LBIST signature after execution of LBIST by SSW with LBISTCTRL1.BODY=0	08E8 <sub>H</sub>	
SCU_LBISTCTRL3_BODY1	SCU_LBISTCTRL3_BODY1 - LBIST signature after execution of LBIST by SSW with LBISTCTRL1.BODY=1	08EC <sub>H</sub>	
SCU_LBISTCTRL0	SCU_LBISTCTRL0 - Value used by SSW when executing LBIST (called "Configuration A")	0960 <sub>H</sub>	
SCU_LBISTCTRL1	SCU_LBISTCTRL1 - Value used by SSW when executing LBIST (called "Configuration A"). BODY determined by HWCFG[6].	0964 <sub>H</sub>	
SCU_LBISTCTRL2	SCU_LBISTCTRL2 - Value used by SSW when executing LBIST (called "Configuration A")	0968 <sub>H</sub>	
CONFIRMATION_ORIG	UCB_SSW_CODE_ORIG	09F0 <sub>H</sub>	
WORD125	RESERVED	09F4 <sub>H</sub>	
CONFIRMATION_COPY	UCB_SSW_CODE_COPY	09F8 <sub>H</sub>	
WORD127	RESERVED	09FC <sub>H</sub>	

### 6.8.2.4 UCB\_USER

This UCB contains data supplied by IFX to the customer application. The entries are documented in the respective chapters (e.g. EDSADC and RIF) or will be documented when necessary (e.g. UID and BOM).

**Table 222 Register Overview - UCB05 (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
UID	UCB_USER_UID - Unique Chip Identifier	0A00 <sub>H</sub>	
UID	UCB_USER_UID - Unique Chip Identifier	0A04 <sub>H</sub>	
UID	UCB_USER_UID - Unique Chip Identifier	0A08 <sub>H</sub>	
UID	UCB_USER_UID - Unique Chip Identifier	0A0C <sub>H</sub>	
BOM	UCB_USER_BOM - Bill of Materials: mold, bond wire material, ball material	0A10 <sub>H</sub>	
EVADC_G00_VDDK	EVADC_G00_VDDK - Device specific values G00: DVDDK/VDDKC	0A20 <sub>H</sub>	195
EVADC_G00_RES	EVADC_G00_RES - Device specific values G00: Reserved	0A24 <sub>H</sub>	
EVADC_G01_VDDK	EVADC_G01_VDDK - Device specific values G01: DVDDK/VDDKC	0A28 <sub>H</sub>	

**Table 222 Register Overview - UCB05 (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Page Number</b>
EVADC_G01_RES	EVADC_G01_RES - Device specific values G01: Reserved	0A2C <sub>H</sub>	
EVADC_G02_VDDK	EVADC_G02_VDDK - Device specific values G02: DVDDK/VDDKC	0A30 <sub>H</sub>	
EVADC_G02_RES	EVADC_G02_RES - Device specific values G02: Reserved	0A34 <sub>H</sub>	
EVADC_G03_VDDK	EVADC_G03_VDDK - Device specific values G03: DVDDK/VDDKC	0A38 <sub>H</sub>	
EVADC_G03_RES	EVADC_G03_RES - Device specific values G03: Reserved	0A3C <sub>H</sub>	
EVADC_G04_VDDK	EVADC_G04_VDDK - Device specific values G04: DVDDK/VDDKC	0A40 <sub>H</sub>	
EVADC_G04_RES	EVADC_G04_RES - Device specific values G04: Reserved	0A44 <sub>H</sub>	
EVADC_G05_VDDK	EVADC_G05_VDDK - Device specific values G05: DVDDK/VDDKC	0A48 <sub>H</sub>	
EVADC_G05_RES	EVADC_G05_RES - Device specific values G05: Reserved	0A4C <sub>H</sub>	
EVADC_G06_VDDK	EVADC_G06_VDDK - Device specific values G06: DVDDK/VDDKC	0A50 <sub>H</sub>	
EVADC_G06_RES	EVADC_G06_RES - Device specific values G06: Reserved	0A54 <sub>H</sub>	
EVADC_G07_VDDK	EVADC_G07_VDDK - Device specific values G07: DVDDK/VDDKC	0A58 <sub>H</sub>	
EVADC_G07_RES	EVADC_G07_RES - Device specific values G07: Reserved	0A5C <sub>H</sub>	
EVADC_G08_VDDK	EVADC_G08_VDDK - Device specific values G08: DVDDK/VDDKC	0A60 <sub>H</sub>	
EVADC_G08_RES	EVADC_G08_RES - Device specific values G08: Reserved	0A64 <sub>H</sub>	
EVADC_G09_VDDK	EVADC_G09_VDDK - Device specific values G09: DVDDK/VDDKC	0A68 <sub>H</sub>	
EVADC_G09_RES	EVADC_G09_RES - Device specific values G09: Reserved	0A6C <sub>H</sub>	
EVADC_G10_VDDK	EVADC_G10_VDDK - Device specific values G10: DVDDK/VDDKC	0A70 <sub>H</sub>	
EVADC_G10_RES	EVADC_G10_RES - Device specific values G10: Reserved	0A74 <sub>H</sub>	
EVADC_G11_VDDK	EVADC_G11_VDDK - Device specific values G11: DVDDK/VDDKC	0A78 <sub>H</sub>	
EVADC_G11_RES	EVADC_G11_RES - Device specific values G11: Reserved	0A7C <sub>H</sub>	
EDSADC_CH00_RES	EDSADC_CH00_RES - Device specific values CH00: Reserved	0A80 <sub>H</sub>	
EDSADC_CH00_IRMS	EDSADC_CH00_IRMS - Device specific values CH00: IRMS	0A84 <sub>H</sub>	195
EDSADC_CH01_RES	EDSADC_CH01_RES - Device specific values CH01: Reserved	0A88 <sub>H</sub>	
EDSADC_CH01_IRMS	EDSADC_CH01_IRMS - Device specific values CH01: IRMS	0A8C <sub>H</sub>	
EDSADC_CH02_RES	EDSADC_CH02_RES - Device specific values CH02: Reserved	0A90 <sub>H</sub>	
EDSADC_CH02_IRMS	EDSADC_CH02_IRMS - Device specific values CH02: IRMS	0A94 <sub>H</sub>	
EDSADC_CH03_RES	EDSADC_CH03_RES - Device specific values CH03: Reserved	0A98 <sub>H</sub>	
EDSADC_CH03_IRMS	EDSADC_CH03_IRMS - Device specific values CH03: IRMS	0A9C <sub>H</sub>	

**Table 222 Register Overview - UCB05 (ascending Offset Address) (cont'd)**

Short Name	Long Name	Offset Address	Page Number
EDSADC_CH04_RES	EDSADC_CH04_RES - Device specific values CH04: Reserved	0AA0 <sub>H</sub>	
EDSADC_CH04_IRMS	EDSADC_CH04_IRMS - Device specific values CH04: IRMS	0AA4 <sub>H</sub>	
EDSADC_CH05_RES	EDSADC_CH05_RES - Device specific values CH05: Reserved	0AA8 <sub>H</sub>	
EDSADC_CH05_IRMS	EDSADC_CH05_IRMS - Device specific values CH05: IRMS	0AAC <sub>H</sub>	
EDSADC_CH06_RES	EDSADC_CH06_RES - Device specific values CH06: Reserved	0AB0 <sub>H</sub>	
EDSADC_CH06_IRMS	EDSADC_CH06_IRMS - Device specific values CH06: IRMS	0AB4 <sub>H</sub>	
EDSADC_CH07_RES	EDSADC_CH07_RES - Device specific values CH07: Reserved	0AB8 <sub>H</sub>	
EDSADC_CH07_IRMS	EDSADC_CH07_IRMS - Device specific values CH07: IRMS	0ABC <sub>H</sub>	
EDSADC_CH08_RES	EDSADC_CH08_RES - Device specific values CH08: Reserved	0AC0 <sub>H</sub>	
EDSADC_CH08_IRMS	EDSADC_CH08_IRMS - Device specific values CH08: IRMS	0AC4 <sub>H</sub>	
EDSADC_CH09_RES	EDSADC_CH09_RES - Device specific values CH09: Reserved	0AC8 <sub>H</sub>	
EDSADC_CH09_IRMS	EDSADC_CH09_IRMS - Device specific values CH09: IRMS	0ACC <sub>H</sub>	
EDSADC_CH10_RES	EDSADC_CH10_RES - Device specific values CH10: Reserved	0AD0 <sub>H</sub>	
EDSADC_CH10_IRMS	EDSADC_CH10_IRMS - Device specific values CH10: IRMS	0AD4 <sub>H</sub>	
EDSADC_CH11_RES	EDSADC_CH11_RES - Device specific values CH11: Reserved	0AD8 <sub>H</sub>	
EDSADC_CH11_IRMS	EDSADC_CH11_IRMS - Device specific values CH11: IRMS	0ADC <sub>H</sub>	
EDSADC_CH12_RES	EDSADC_CH12_RES - Device specific values CH12: Reserved	0AE0 <sub>H</sub>	
EDSADC_CH12_IRMS	EDSADC_CH12_IRMS - Device specific values CH12: IRMS	0AE4 <sub>H</sub>	
EDSADC_CH13_RES	EDSADC_CH13_RES - Device specific values CH13: Reserved	0AE8 <sub>H</sub>	
EDSADC_CH13_IRMS	EDSADC_CH13_IRMS - Device specific values CH13: IRMS	0AEC <sub>H</sub>	
RIF_LVDSCON1	RIF_LVDSCON1 - Trimming value in bits [2:0] for RIF0_LVDSCON1.RTERM and in bits [18:16] for RIF1_LVDSCON1.RTERM	0AFC <sub>H</sub>	196
CONFIRMATION_ORIG	UCB_USER_CODE_ORIG	0BF0 <sub>H</sub>	
CONFIRMATION_COPY	UCB_USER_CODE_COPY	0BF8 <sub>H</sub>	

### 6.8.2.5 UCB\_RETEST

Reserved by IFX, and not usable by customer.

### 6.8.2.6 UCB\_PFLASH\_ORIG and UCB\_PFLASH\_COPY

The UCB\_PFLASH (ORIG and COPY) contain the user defined PFlash protection. It is protected with the password PW0 to PW7. The protection configuration is transferred into the registers DMU\_HP\_PROCONPps (with “p” looping over the implemented PFlash banks and “s” looping over groups of 32 sectors) and DMU\_HF\_PROCONPF (see DMU chapter).

### Delivery State

The “CONFIRMATION” code entries contain the UNLOCKED value (see “DMU” chapter, section “UCB Confirmation”). All other addresses are delivered erased.

**Table 223 Register Overview - UCB16 (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
PROCONPp0	DMU_HP_PROCONPp0 - PFp protection for logical sectors S0 - S31	2000 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONPp1	DMU_HP_PROCONPp1 - PFp protection for logical sectors S32 - S63	2004 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONPp2	DMU_HP_PROCONPp2 - PFp protection for logical sectors S64 - S95	2008 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONPp3	DMU_HP_PROCONPp3 - PFp protection for logical sectors S96 - S127	200C <sub>H</sub> +p*20 <sub>H</sub>	
PROCONPp4	DMU_HP_PROCONPp4 - PFp protection for logical sectors S128 - S159	2010 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONPp5	DMU_HP_PROCONPp5 - PFp protection for logical sectors S160 - S191	2014 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONPF	DMU_HF_PROCONPF	20C0 <sub>H</sub>	
PW0	UCB_PFLASH_ORIG_PW	2100 <sub>H</sub>	
PW1	UCB_PFLASH_ORIG_PW	2104 <sub>H</sub>	
PW2	UCB_PFLASH_ORIG_PW	2108 <sub>H</sub>	
PW3	UCB_PFLASH_ORIG_PW	210C <sub>H</sub>	
PW4	UCB_PFLASH_ORIG_PW	2110 <sub>H</sub>	
PW5	UCB_PFLASH_ORIG_PW	2114 <sub>H</sub>	
PW6	UCB_PFLASH_ORIG_PW	2118 <sub>H</sub>	
PW7	UCB_PFLASH_ORIG_PW	211C <sub>H</sub>	
CONFIRMATION	UCB_PFLASH_ORIG_CODE	21F0 <sub>H</sub>	

The layout of UCB24 is identical to UCB16 but on different offset addresses (see [Table 218](#)) and is therefore not shown here.

### 6.8.2.7 UCB\_DFLASH\_ORIG and UCB\_DFLASH\_COPY

The UCB\_DFLASH (ORIG and COPY) contain the user defined DFlash protection. It is protected with the password PW0 to PW7. The protection configuration is transferred into the registers DMU\_HF\_PROCONUSR, DMU\_HF\_PROCONDF and DMU\_HF\_PROCONRAM (see DMU chapter).

### Delivery State

The “CONFIRMATION” code entries contain the UNLOCKED value (see “DMU” chapter, section “UCB Confirmation”). All other addresses are delivered erased.

**Table 224 Register Overview - UCB17 (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
PROCONUSR	DMU_HF_PROCONUSR	2200 <sub>H</sub>	
PROCOND <sub>F</sub>	DMU_HF_PROCOND <sub>F</sub>	2204 <sub>H</sub>	
PROCONRAM	DMU_HF_PROCONRAM	2208 <sub>H</sub>	
PW0	UCB_DFLASH_ORIG_PW	2300 <sub>H</sub>	
PW1	UCB_DFLASH_ORIG_PW	2304 <sub>H</sub>	
PW2	UCB_DFLASH_ORIG_PW	2308 <sub>H</sub>	
PW3	UCB_DFLASH_ORIG_PW	230C <sub>H</sub>	
PW4	UCB_DFLASH_ORIG_PW	2310 <sub>H</sub>	
PW5	UCB_DFLASH_ORIG_PW	2314 <sub>H</sub>	
PW6	UCB_DFLASH_ORIG_PW	2318 <sub>H</sub>	
PW7	UCB_DFLASH_ORIG_PW	231C <sub>H</sub>	
CONFIRMATION	UCB_DFLASH_ORIG_CODE	23F0 <sub>H</sub>	

The layout of UCB25 is identical to UCB17 but on different offset addresses (see [Table 218](#)) and therefore not shown here.

### 6.8.2.8 UCB\_DBG\_ORIG and UCB\_DBG\_COPY

The UCB\_DBG (ORIG and COPY) contain the user defined debug protection. It is protected with the password PW0 to PW7. The protection configuration is transferred into the register DMU\_HF\_PROCOND<sub>BG</sub> (see DMU chapter).

#### Delivery State

The “CONFIRMATION” code entries contain the UNLOCKED value (see “DMU” chapter, section “UCB Confirmation”). All other addresses are delivered erased.

**Table 225 Register Overview - UCB18 (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
PROCOND <sub>BG</sub>	DMU_HF_PROCOND <sub>BG</sub>	2400 <sub>H</sub>	
PW0	UCB_DBG_ORIG_PW	2500 <sub>H</sub>	
PW1	UCB_DBG_ORIG_PW	2504 <sub>H</sub>	
PW2	UCB_DBG_ORIG_PW	2508 <sub>H</sub>	
PW3	UCB_DBG_ORIG_PW	250C <sub>H</sub>	
PW4	UCB_DBG_ORIG_PW	2510 <sub>H</sub>	
PW5	UCB_DBG_ORIG_PW	2514 <sub>H</sub>	
PW6	UCB_DBG_ORIG_PW	2518 <sub>H</sub>	
PW7	UCB_DBG_ORIG_PW	251C <sub>H</sub>	
CONFIRMATION	UCB_DBG_ORIG_CODE	25F0 <sub>H</sub>	

The layout of UCB26 is identical to UCB18 but on different offset addresses (see [Table 218](#)) and therefore not shown here.

### 6.8.2.9 UCB\_HSM\_ORIG and UCB\_HSM\_COPY

The UCB\_HSM (ORIG and COPY) contain the user defined HSM configuration. It is transferred into the register DMU\_SP\_PROCONHSM (see DMU chapter).

#### Delivery State

The “CONFIRMATION” code entries contain the UNLOCKED value (see “DMU” chapter, section “UCB Confirmation”). All other addresses are delivered erased.

**Table 226 Register Overview - UCB19 (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
PROCONHSM	DMU_SP_PROCONHSM	2600 <sub>H</sub>	
CONFIRMATION	UCB_HSM_ORIG_CODE	27F0 <sub>H</sub>	

The layout of UCB27 is identical to UCB19 but on different offset addresses (see [Table 218](#)) and therefore not shown here.

### 6.8.2.10 UCB\_HSMCOTP0/1\_ORIG and UCB\_HSMCOTP0/1\_COPY

The UCB\_HSMCOTP (two sets 0 and 1 each with ORIG and COPY) contain the OTP part of the user defined HSM protection and configuration. It is transferred into the registers DMU\_SF\_PROCONUSR, DMU\_SP\_PROCONHSMC\* (see DMU chapter).

#### Delivery State

The “CONFIRMATION” code entries contain the UNLOCKED value (see “DMU” chapter, section “UCB Confirmation”). All other addresses are delivered erased.

**Table 227 Register Overview - UCB20 (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
PROCONUSR	DMU_SF_PROCONUSR	2800 <sub>H</sub>	
PROCONHSMCBS	DMU_SP_PROCONHSMCBS	2804 <sub>H</sub>	
PROCONHSMCX0	DMU_SP_PROCONHSMCX0	2808 <sub>H</sub>	
PROCONHSMCX1	DMU_SP_PROCONHSMCX1	280C <sub>H</sub>	
PROCONHSMCOTP0	DMU_SP_PROCONHSMCOTP0	2810 <sub>H</sub>	
PROCONHSMCOTP1	DMU_SP_PROCONHSMCOTP1	2814 <sub>H</sub>	
PROCONHSMCFG	DMU_SP_PROCONHSMCFG	2818 <sub>H</sub>	
CONFIRMATION	UCB_HSMCOTP0_ORIG_CODE	29F0 <sub>H</sub>	

The layout of UCB21, UCB28 and UCB29 is identical to UCB20 but on different offset addresses (see [Table 218](#)) and therefore not shown here.

### 6.8.2.11 UCB\_ECPRIORIG\_ORIG and UCB\_ECPRIORIG\_COPY

The UCB\_ECPRIORIG (ORIG and COPY) contain the user defined erase counter configuration. It is protected with the password PW0 to PW7. The configuration is transferred into the registers DMU\_HP\_ECPRIOps (with “p” looping over the implemented PFlash banks and “s” looping over groups of 32 sectors, see DMU chapter).

#### Delivery State

The “CONFIRMATION” code entries contain the UNLOCKED value (see “DMU” chapter, section “UCB Confirmation”). All other addresses are delivered erased.

**Table 228 Register Overview - UCB22 (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
ECPRIOp0	DMU_HP_ECPRIOp0 - PFp Erase Counter priority for logical sectors S0 - S31	2C00 <sub>H</sub> +p*20 <sub>H</sub>	
ECPRIOp1	DMU_HP_ECPRIOp1 - PFp Erase Counter priority for logical sectors S32 - S63	2C04 <sub>H</sub> +p*20 <sub>H</sub>	
ECPRIOp2	DMU_HP_ECPRIOp2 - PFp Erase Counter priority for logical sectors S64 - S95	2C08 <sub>H</sub> +p*20 <sub>H</sub>	
ECPRIOp3	DMU_HP_ECPRIOp3 - PFp Erase Counter priority for logical sectors S96 - S127	2C0C <sub>H</sub> +p*20 <sub>H</sub>	
ECPRIOp4	DMU_HP_ECPRIOp4 - PFp Erase Counter priority for logical sectors S128 - S159	2C10 <sub>H</sub> +p*20 <sub>H</sub>	
ECPRIOp5	DMU_HP_ECPRIOp5 - PFp Erase Counter priority for logical sectors S160 - S191	2C14 <sub>H</sub> +p*20 <sub>H</sub>	
PW0	UCB_ECPRIORIG_PW	2D00 <sub>H</sub>	
PW1	UCB_ECPRIORIG_PW	2D04 <sub>H</sub>	
PW2	UCB_ECPRIORIG_PW	2D08 <sub>H</sub>	
PW3	UCB_ECPRIORIG_PW	2D0C <sub>H</sub>	
PW4	UCB_ECPRIORIG_PW	2D10 <sub>H</sub>	
PW5	UCB_ECPRIORIG_PW	2D14 <sub>H</sub>	
PW6	UCB_ECPRIORIG_PW	2D18 <sub>H</sub>	
PW7	UCB_ECPRIORIG_PW	2D1C <sub>H</sub>	
CONFIRMATION	UCB_ECPRIORIG_CODE	2DF0 <sub>H</sub>	

The layout of UCB30 is identical to UCB22 but on different offset addresses (see [Table 218](#)) and therefore not shown here.

### 6.8.2.12 UCB\_SWAP\_ORIG and UCB\_SWAP\_COPY

The UCB\_SWAP (ORIG and COPY) contain the user defined SWAP configuration. It is protected with the password PW0 to PW7. The SWAP configuration is evaluated by the SSW (see “Software update Over The Air (SOTA) chapter”).

**Delivery State**

The “CONFIRMATION” code entries (not CONFIRMATIONLx and CONFIRMATIONHx entries!) contain the UNLOCKED value (see “DMU” chapter, section “UCB Confirmation”). All other addresses are delivered erased.

**Table 229 Register Overview - UCB23 (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
MARKERLx	UCB_SWAP_ORIG_MARKERLx	2E00 <sub>H</sub> +x*10 <sub>H</sub>	<a href="#">196</a>
MARKERHx	UCB_SWAP_ORIG_MARKERHx	2E04 <sub>H</sub> +x*10 <sub>H</sub>	<a href="#">197</a>
CONFIRMATIONLx	UCB_SWAP_ORIG_CONFIRMATIONLx	2E08 <sub>H</sub> +x*10 <sub>H</sub>	<a href="#">197</a>
CONFIRMATIONHx	UCB_SWAP_ORIG_CONFIRMATIONHx	2E0C <sub>H</sub> +x*10 <sub>H</sub>	<a href="#">198</a>
PW0	UCB_SWAP_ORIG_PW	2F00 <sub>H</sub>	
PW1	UCB_SWAP_ORIG_PW	2F04 <sub>H</sub>	
PW2	UCB_SWAP_ORIG_PW	2F08 <sub>H</sub>	
PW3	UCB_SWAP_ORIG_PW	2F0C <sub>H</sub>	
PW4	UCB_SWAP_ORIG_PW	2F10 <sub>H</sub>	
PW5	UCB_SWAP_ORIG_PW	2F14 <sub>H</sub>	
PW6	UCB_SWAP_ORIG_PW	2F18 <sub>H</sub>	
PW7	UCB_SWAP_ORIG_PW	2F1C <sub>H</sub>	
CONFIRMATION	UCB_SWAP_ORIG_CODE	2FF0 <sub>H</sub>	

The layout of UCB31 is identical to UCB23 but on different offset addresses (see [Table 218](#)) and therefore not shown here.

**6.8.2.13 UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY (y = 0 - 7)**

The UCB\_OTP (8 sets “y” with each ORIG and COPY) contain the user defined PFlash OTP and WOP protection. It is transferred into the registers DMU\_HP\_PROCONOTPps, DMU\_HP\_PROCONWOPps (with “p” looping over the implemented PFlash banks and “s” looping over groups of 32 sectors) and DMU\_HP\_PROCONTP.

**Delivery State**

The “CONFIRMATION” code entries contain the UNLOCKED value (see “DMU” chapter, section “UCB Confirmation”). All other addresses are delivered erased.

**Table 230 Register Overview - UCB32 (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
PROCONOTPp0	DMU_HP_PROCONOTPp0 - Set 0 of PFp OTP protection for logical sectors S0 - S31	4000 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONOTPp1	DMU_HP_PROCONOTPp1 - Set 0 of PFp OTP protection for logical sectors S32 - S63	4004 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONOTPp2	DMU_HP_PROCONOTPp2 - Set 0 of PFp OTP protection for logical sectors S64 - S95	4008 <sub>H</sub> +p*20 <sub>H</sub>	

**Table 230 Register Overview - UCB32 (ascending Offset Address) (cont'd)**

Short Name	Long Name	Offset Address	Page Number
PROCONOTPp3	DMU_HP_PROCONOTPp3 - Set 0 of PFp OTP protection for logical sectors S96 - S127	400C <sub>H</sub> +p*20 <sub>H</sub>	
PROCONOTPp4	DMU_HP_PROCONOTPp4 - Set 0 of PFp OTP protection for logical sectors S128 - S159	4010 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONOTPp5	DMU_HP_PROCONOTPp5 - Set 0 of PFp OTP protection for logical sectors S160 - S191	4014 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONWOPp0	DMU_HP_PROCONWOPp0 - Set 0 of PFp WOP protection for logical sectors S0 - S31	4100 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONWOPp1	DMU_HP_PROCONWOPp1 - Set 0 of PFp WOP protection for logical sectors S32 - S63	4104 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONWOPp2	DMU_HP_PROCONWOPp2 - Set 0 of PFp WOP protection for logical sectors S64 - S95	4108 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONWOPp3	DMU_HP_PROCONWOPp3 - Set 0 of PFp WOP protection for logical sectors S96 - S127	410C <sub>H</sub> +p*20 <sub>H</sub>	
PROCONWOPp4	DMU_HP_PROCONWOPp4 - Set 0 of PFp WOP protection for logical sectors S128 - S159	4110 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONWOPp5	DMU_HP_PROCONWOPp5 - Set 0 of PFp WOP protection for logical sectors S160 - S191	4114 <sub>H</sub> +p*20 <sub>H</sub>	
PROCONTp	DMU_HF_PROCONTp - Tuning Protection configuration set 0	41E8 <sub>H</sub>	
CONFIRMATION	UCB_OTP0_ORIG_CODE	41F0 <sub>H</sub>	

The layout of UCB33 to UCB47 is identical to UCB32 but on different offset addresses (see [Table 218](#)) and therefore not shown here.

### 6.8.2.14 UCB\_REDSEC

This UCB contains the redundancy activation for the Flash banks. In case certain redundancy is already used by IFX during the test flow these entries are marked as “USED”. Physically not available redundancy is marked as “FAILED”. The available entries can be used to extend the life time of the device. See DMU chapter for details.

#### Delivery State

The “CONFIRMATION\_ORIG” and “CONFIRMATION\_COPY” code entries contain the CONFIRMED value (see “DMU” chapter, section “UCB Confirmation”). As described above the remaining entries can be device specific.

**Table 231 Register Overview - UCB12 (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
REDSECPLps	UCB_REDSECPLps - Low Word of Wordline Redundancy for PFlash p Redundant Sector s	1800 <sub>H</sub> +p*40 <sub>H</sub> +s*8	<a href="#">198</a>
REDSECPHps	UCB_REDSECPHps - High Word of Wordline Redundancy for PFlash p Redundant Sector s	1804 <sub>H</sub> +p*40 <sub>H</sub> +s*8	<a href="#">199</a>
REDSECDL0s	UCB_REDSECDL0s - Low Word of Wordline Redundancy for DFlash0 Redundant Sector s	1980 <sub>H</sub> +s*8	<a href="#">199</a>

**Table 231 Register Overview - UCB12 (ascending Offset Address) (cont'd)**

Short Name	Long Name	Offset Address	Page Number
REDSECDH0s	UCB_REDSECDH0s - High Word of Wordline Redundancy for DFlash0 Redundant Sector s	1984 <sub>H</sub> +s*8	<a href="#">200</a>
REDSECDL1s	UCB_REDSECDL1s - Low Word of Wordline Redundancy for DFlash1 Redundant Sector s	19C0 <sub>H</sub> +s*8	<a href="#">201</a>
REDSECDH1s	UCB_REDSECDH1s - High Word of Wordline Redundancy for DFlash1 Redundant Sector s	19C4 <sub>H</sub> +s*8	<a href="#">201</a>
CONFIRMATION_ORIG	UCB_REDSEC_CODE_ORIG	19F0 <sub>H</sub>	
CONFIRMATION_COPY	UCB_REDSEC_CODE_COPY	19F8 <sub>H</sub>	

## 6.8.3 UCB Entries

UCB entries that relate directly to registers in hardware are not documented here as the register layout can be found in the respective module chapter. The following layouts are data structures without direct hardware match.

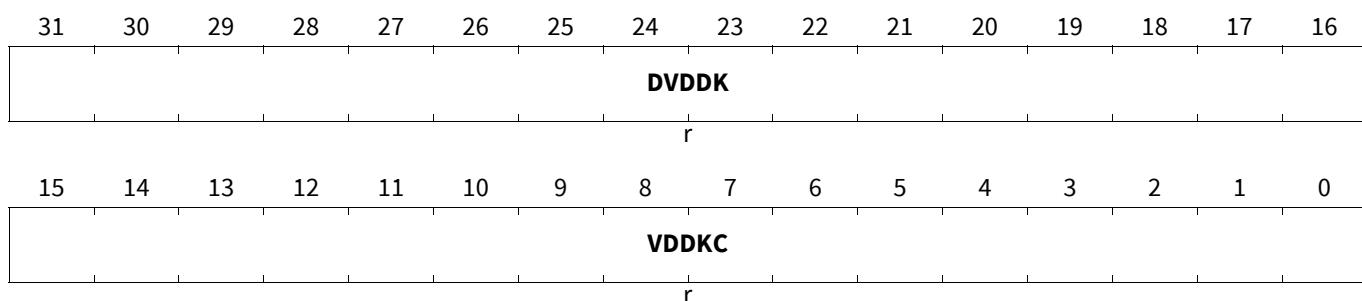
### 6.8.3.1 UCB\_USER

#### EVADC\_G00\_VDDK - Device specific values G00: DVDDK/VDDKC

See EVADC chapter for a detailed documentation of these UCB entries.

##### EVADC\_G00\_VDDK

**EVADC\_G00\_VDDK - Device specific values G00: DVDDK/VDDKC(0A20<sub>H</sub>)**      **Default Flash Value: 0000 0000<sub>H</sub>**



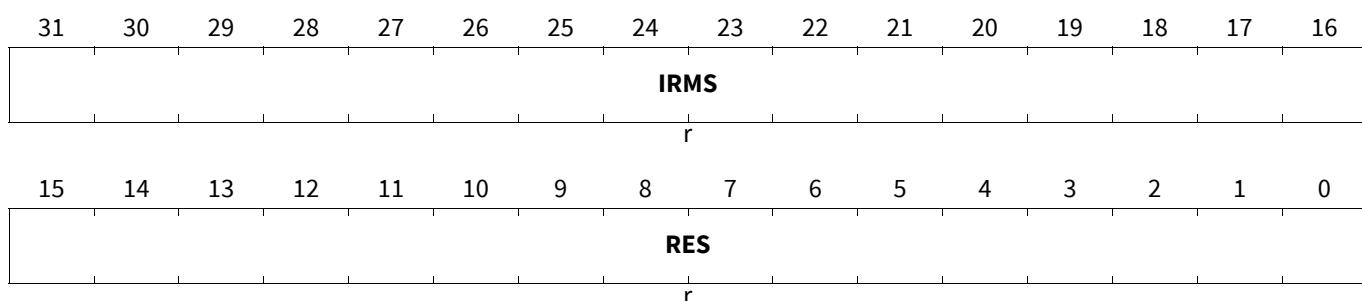
Field	Bits	Type	Description
<b>VDDKC</b>	15:0	r	<b>VDDKC</b> Value of $V_{DDK}$ measured at $-40^{\circ}\text{C}$ in [mV]
<b>DVDDK</b>	31:16	r	<b>DVDDK</b> Temperature deviation DVDDK in [ $\mu\text{V/K}$ ] measured at $V_{DDM}$ of 4.5 V

#### EDSADC\_CH00\_IRMS - Device specific values CH00: IRMS

See EDSADC chapter for a detailed documentation of these UCB entries.

##### EDSADC\_CH00\_IRMS

**EDSADC\_CH00\_IRMS - Device specific values CH00: IRMS(0A84<sub>H</sub>)**      **Default Flash Value: 0000 0000<sub>H</sub>**



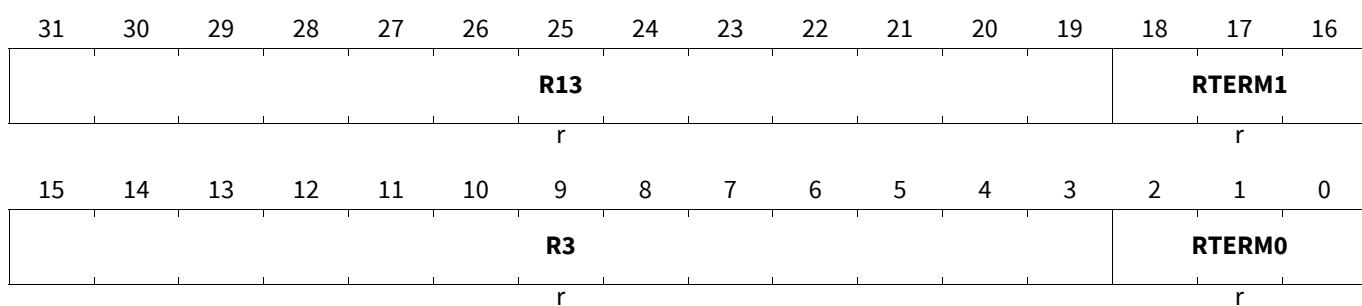
Field	Bits	Type	Description
<b>RES</b>	15:0	r	<b>Reserved</b>
<b>IRMS</b>	31:16	r	<b>IRMS</b> $I_{RMS}$ in [0.01 $\mu\text{A}$ ] measured at 5 V and $f_{MOD}$ of 26.67 MHz

**RIF\_LVDSCON1 - Trimming value in bits [2:0] for RIF0\_LVDSCON1.RTERM and in bits [18:16] for RIF1\_LVDSCON1.RTERM**

The trimming values for the RTERM bitfields of RIF0\_LVDSCON1 and RIF1\_LVDSCON1 are combined in this UCB entry.

**RIF\_LVDSCON1**

**RIF\_LVDSCON1 - Trimming value in bits [2:0] for RIF0\_LVDSCON1.RTERM and in bits [18:16] for RIF1\_LVDSCON1.RTERM** **(0AFC<sub>H</sub>)** **Default Flash Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
RTERM0	2:0	r	<b>RTERM - Termination resistor configuration for RIF0_LVDSCON1 register.</b> Value to be copied into RIF0_LVDSCON1.RTERM register field.
R3	15:3	r	<b>Reserved - RES</b>
RTERM1	18:16	r	<b>RTERM - Termination resistor configuration for RIF1_LVDSCON1 register.</b> Value to be copied into RIF1_LVDSCON1.RTERM register field.
R13	31:19	r	<b>Reserved - RES</b>

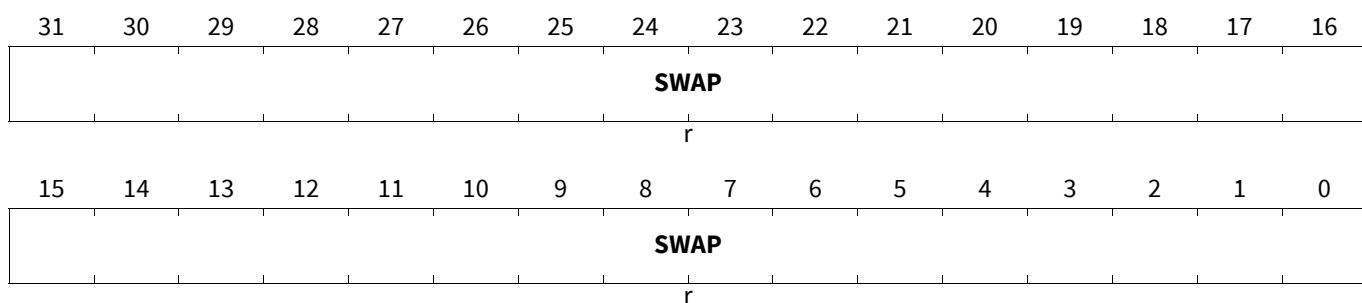
### 6.8.3.2 UCB\_SWAP\_ORIG and UCB\_SWAP\_COPY

**UCB\_SWAP\_ORIG\_MARKERLx**

Determines the system address map used by the current running application. For more details please refer to the "Software update Over The Air(SOTA)" chapter.

**MARKERLx (x=0-15)**

**UCB\_SWAP\_ORIG\_MARKERLx** **(2E00<sub>H</sub>+x\*10<sub>H</sub>)** **Default Flash Value: 0000 0000<sub>H</sub>**



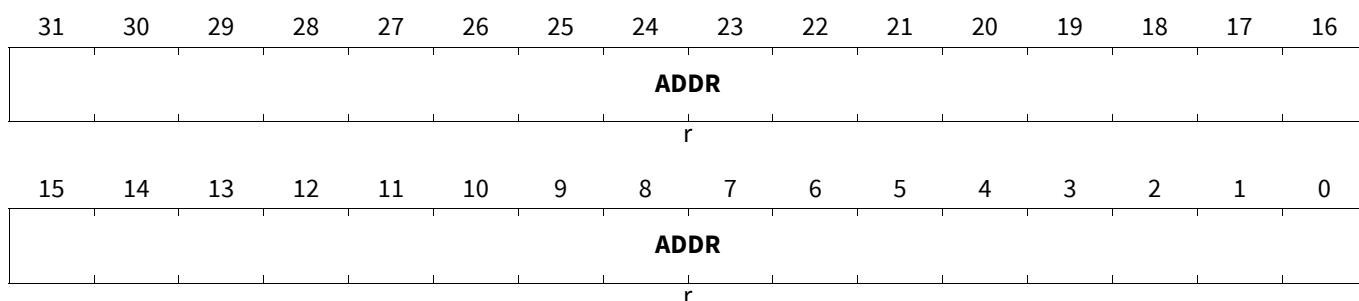
Field	Bits	Type	Description
<b>SWAP</b>	31:0	r	<b>SWAP</b> 00000000 <sub>H</sub> <b>ERASED</b> , Erased state 00000055 <sub>H</sub> <b>STD</b> , Selects standard address map 000000AA <sub>H</sub> <b>ALT</b> , Selects alternate address map

**UCB\_SWAP\_ORIG\_MARKERHx**

Holds the 32-bit system address of the corresponding MARKERLx.SWAP UCB\_SWAP entry confirming its validity. For safety purposes this is checked by the startup software before the address map is installed.

**MARKERHx (x=0-15)**

**UCB\_SWAP\_ORIG\_MARKERHx** **(2E04<sub>H</sub>+x\*10<sub>H</sub>)** **Default Flash Value: 0000 0000<sub>H</sub>**



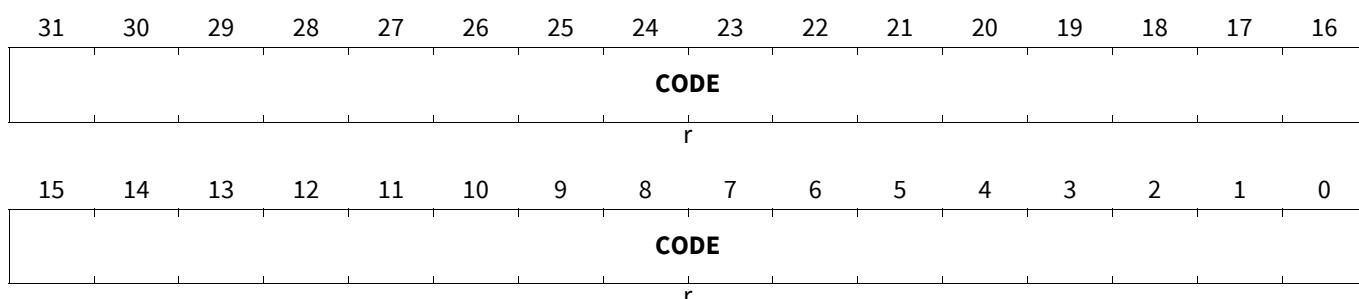
Field	Bits	Type	Description
<b>ADDR</b>	31:0	r	<b>Address of corresponding MARKERLx.SWAP entry as confirmation or erased</b>

**UCB\_SWAP\_ORIG\_CONFIRMATIONLx**

This holds the confirmation code of the address map configured in the previous SWAP MARKERL/H. A valid code indicates that the above configuration of the address map can be installed by the startup software in the SCU\_SWAPCTRL register.

**CONFIRMATIONLx (x=0-15)**

**UCB\_SWAP\_ORIG\_CONFIRMATIONLx** **(2E08<sub>H</sub>+x\*10<sub>H</sub>)** **Default Flash Value: 0000 0000<sub>H</sub>**



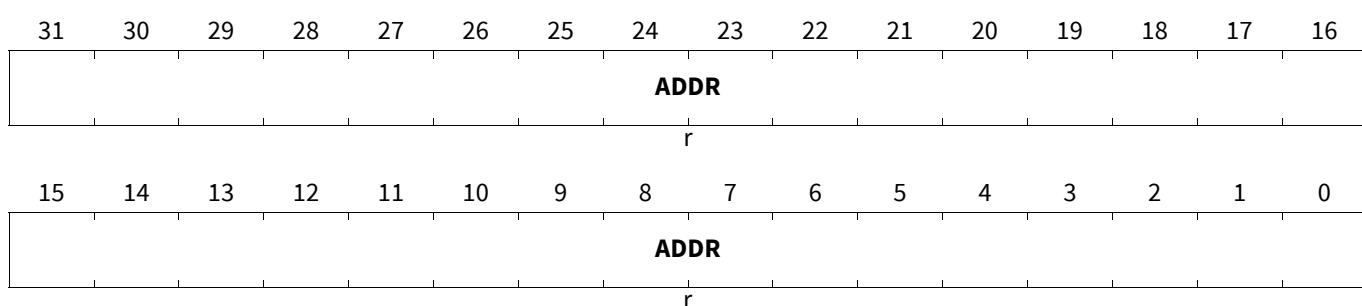
Field	Bits	Type	Description
<b>CODE</b>	31:0	r	<b>CODE</b> 00000000 <sub>H</sub> <b>ERASED</b> , Erased state 57B5327F <sub>H</sub> <b>CONFIRMED</b> , Confirmed code

**UCB\_SWAP\_ORIG\_CONFIRMATIONHx**

Holds the 32-bit system address of the corresponding CONFIRMATIONLx.CODE UCB\_SWAP entry confirming its validity. This is checked by startup software before the MARKER regions are read. The address map is installed only if this address matches the actual address of this location.

**CONFIRMATIONHx (x=0-15)**

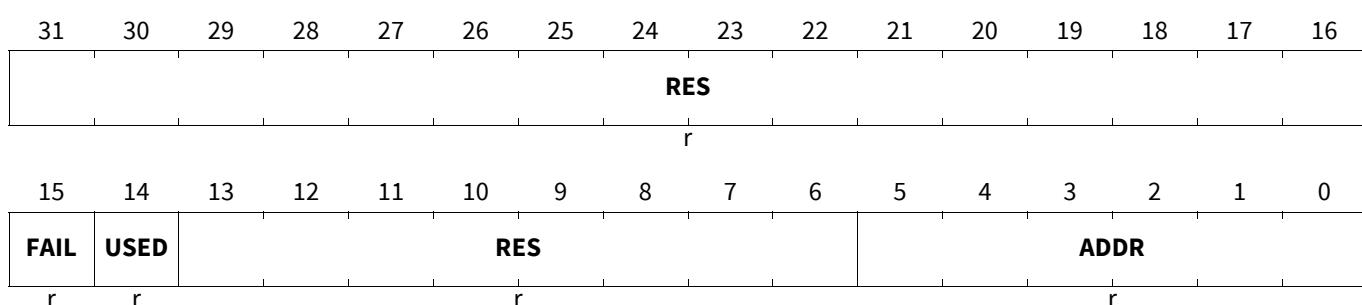
**UCB\_SWAP\_ORIG\_CONFIRMATIONHx**  $(2E0C_H + x * 10_H)$  **Default Flash Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
ADDR	31:0	r	Address of corresponding CONFIRMATIONLx.CODE entry as confirmation or erased

**6.8.3.3 UCB\_REDSEC****UCB\_REDSECPLps - Low Word of Wordline Redundancy for PFlash p Redundant Sector s****REDSECPLps (p=0-5;s=0-7)****UCB\_REDSECPLps - Low Word of Wordline Redundancy for PFlash p Redundant Sector s**

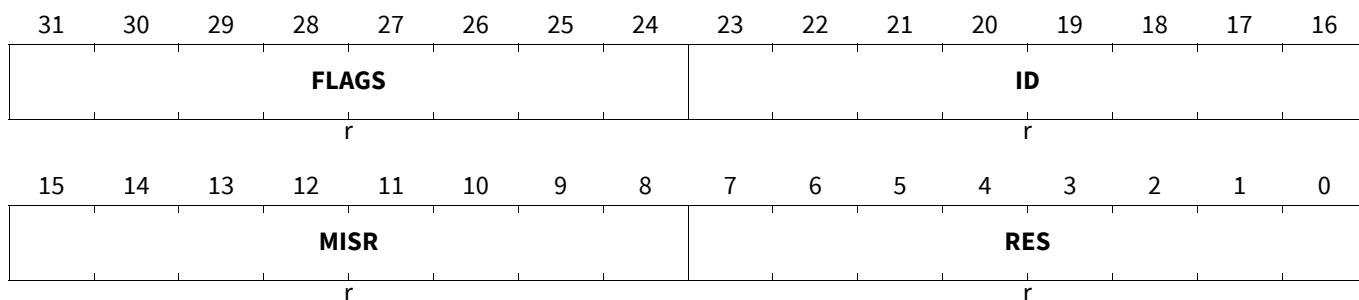
**(1800<sub>H</sub>+p\*40<sub>H</sub>+s\*8)** **Reset Value: Table 232**



Field	Bits	Type	Description
ADDR	5:0	r	Address of defective Sector in 1024KiB block.
RES	13:6, 31:16	r	Reserved
USED	14	r	Used State $0_B$ UNUSED, Entry free $1_B$ USED, Entry used
FAIL	15	r	Failed State $0_B$ OK, Entry available $1_B$ FAIL, Entry unavailable (redundancy not existing or unusable)

**Table 232 Reset Values of REDSECPLps (p=0-5;s=0-7)**

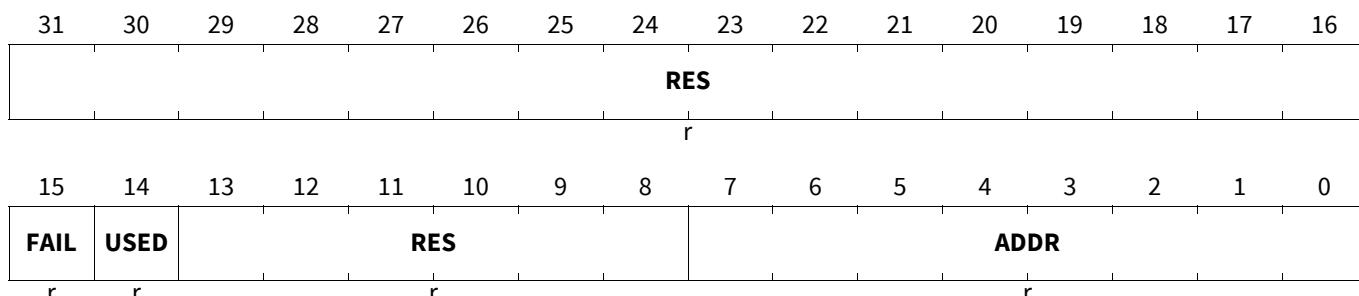
Reset Type	Reset Value	Note
Default Flash	0000 0000 <sub>H</sub>	Available Entry
Default Flash	0000 8000 <sub>H</sub>	Not Available Default Entry

**UCB\_REDSECPHps - High Word of Wordline Redundancy for PFlash p Redundant Sector s****REDSECPHps (p=0-5;s=0-7)****UCB\_REDSECPHps - High Word of Wordline Redundancy for PFlash p Redundant Sector s  
(1804<sub>H</sub>+p\*40<sub>H</sub>+s\*8)      Reset Value: [Table 233](#)**

Field	Bits	Type	Description
<b>RES</b>	7:0	r	<b>Reserved</b>
<b>MISR</b>	15:8	r	<b>MISR Result</b>
<b>ID</b>	23:16	r	<b>Order of Installation</b>
<b>FLAGS</b>	31:24	r	<b>Flags</b> 00 <sub>H</sub> <b>TEST</b> , Entry installed by device test FF <sub>H</sub> <b>RLS</b> , Entry installed by Replace Logical Sector command

**Table 233 Reset Values of REDSECPHps (p=0-5;s=0-7)**

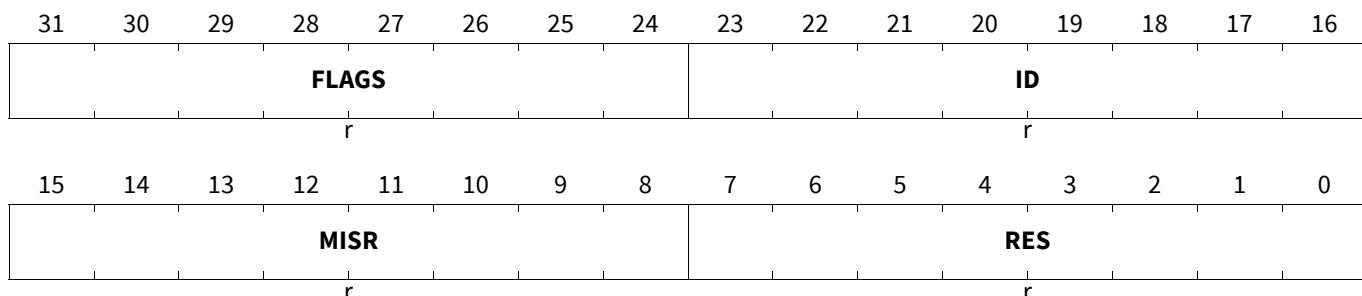
Reset Type	Reset Value	Note
Default Flash	0000 0000 <sub>H</sub>	Available Entry

**UCB\_REDSECDL0s - Low Word of Wordline Redundancy for DFlash0 Redundant Sector s****REDSECDL0s (s=0-7)****UCB\_REDSECDL0s - Low Word of Wordline Redundancy for DFlash0 Redundant Sector s(1980<sub>H</sub>+s\*8)      Reset Value: [Table 234](#)**

Field	Bits	Type	Description
<b>ADDR</b>	7:0	r	<b>Address of defective Sector in 1024KiB block.</b>
<b>RES</b>	13:8, 31:16	r	<b>Reserved</b>
<b>USED</b>	14	r	<b>Used State</b> $0_B$ <b>UNUSED</b> , Entry free $1_B$ <b>USED</b> , Entry used
<b>FAIL</b>	15	r	<b>Failed State</b> $0_B$ <b>OK</b> , Entry available $1_B$ <b>FAIL</b> , Entry unavailable (redundancy not existing or unusable)

**Table 234 Reset Values of REDSECDL0s (s=0-7)**

Reset Type	Reset Value	Note
Default Flash	0000 0000 <sub>H</sub>	Available Entry
Default Flash	0000 8000 <sub>H</sub>	Not Available Default Entry

**UCB\_REDSECDH0s - High Word of Wordline Redundancy for DFlash0 Redundant Sector s****REDSECDH0s (s=0-7)****UCB\_REDSECDH0s - High Word of Wordline Redundancy for DFlash0 Redundant Sector s(1984<sub>H</sub>+s\*8) Reset**Value: [Table 235](#)

Field	Bits	Type	Description
<b>RES</b>	7:0	r	<b>Reserved</b>
<b>MISR</b>	15:8	r	<b>MISR Result</b>
<b>ID</b>	23:16	r	<b>Order of Installation</b>
<b>FLAGS</b>	31:24	r	<b>Flags</b> $00_H$ <b>TEST</b> , Entry installed by device test $FF_H$ <b>RLS</b> , Entry installed by Replace Logical Sector command

**Table 235 Reset Values of REDSECDH0s (s=0-7)**

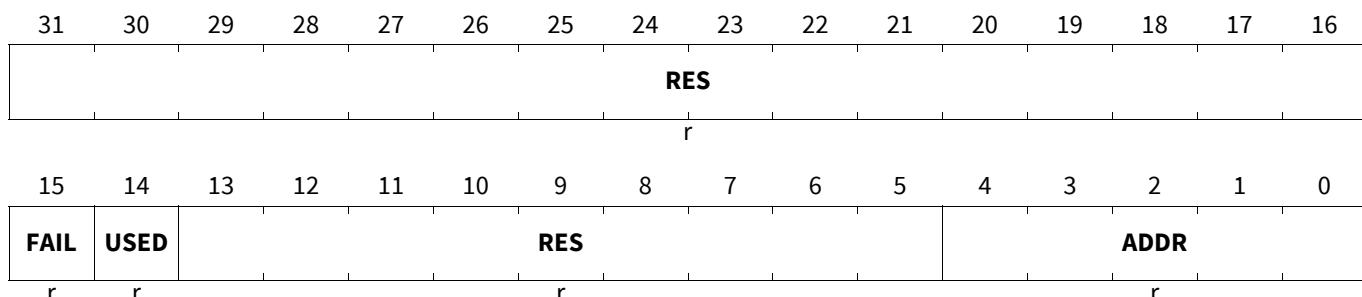
Reset Type	Reset Value	Note
Default Flash	0000 0000 <sub>H</sub>	Available Entry

### UCB\_REDSECDL1s - Low Word of Wordline Redundancy for DFlash1 Redundant Sector s

#### REDSECDL1s (s=0-5)

UCB\_REDSECDL1s - Low Word of Wordline Redundancy for DFlash1 Redundant Sector s( $19C0_H + s * 8$ ) Reset

Value: [Table 236](#)



Field	Bits	Type	Description
<b>ADDR</b>	4:0	r	<b>Address of defective Sector in 1024KiB block.</b>
<b>RES</b>	13:5, 31:16	r	<b>Reserved</b>
<b>USED</b>	14	r	<b>Used State</b> $0_B$ <b>UNUSED</b> , Entry free $1_B$ <b>USED</b> , Entry used
<b>FAIL</b>	15	r	<b>Failed State</b> $0_B$ <b>OK</b> , Entry available $1_B$ <b>FAIL</b> , Entry unavailable (redundancy not existing or unusable)

**Table 236 Reset Values of REDSECDL1s (s=0-5)**

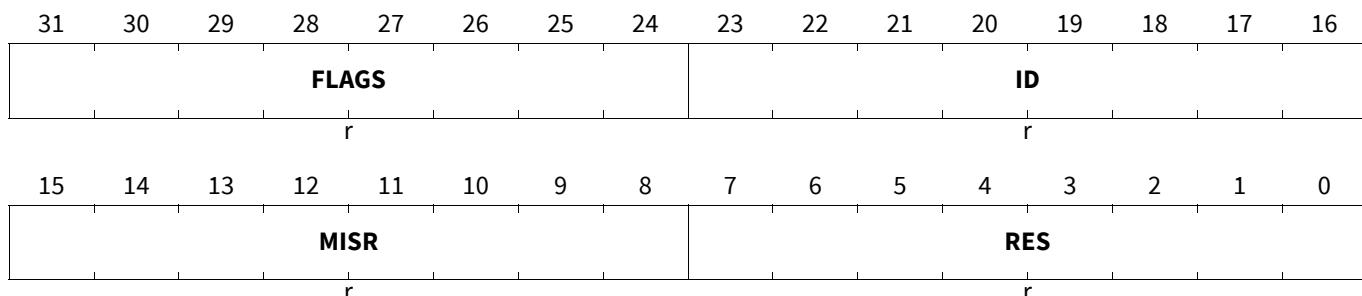
Reset Type	Reset Value	Note
Default Flash	$0000\ 0000_H$	Available Entry
Default Flash	$0000\ 8000_H$	Not Available Default Entry

### UCB\_REDSECDH1s - High Word of Wordline Redundancy for DFlash1 Redundant Sector s

#### REDSECDH1s (s=0-5)

UCB\_REDSECDH1s - High Word of Wordline Redundancy for DFlash1 Redundant Sector s( $19C4_H + s * 8$ ) Reset

Value: [Table 237](#)



Field	Bits	Type	Description
<b>RES</b>	7:0	r	<b>Reserved</b>
<b>MISR</b>	15:8	r	<b>MISR Result</b>
<b>ID</b>	23:16	r	<b>Order of Installation</b>
<b>FLAGS</b>	31:24	r	<b>Flags</b> $00_H$ <b>TEST</b> , Entry installed by device test $FF_H$ <b>RLS</b> , Entry installed by Replace Logical Sector command

**Table 237 Reset Values of REDSECDH1s (s=0-5)**

Reset Type	Reset Value	Note
Default Flash	$0000\ 0000_H$	Available Entry

## 6.8.4 Revision History

**Table 238 Changes from V2.0.20 on**

Reference	Changes to Previous Version	Comment
<b>V2.0.20</b>		
<a href="#">Page 183</a> , <a href="#">187</a> , <a href="#">188</a> , <a href="#">189</a> , <a href="#">190</a> , <a href="#">190</a> , <a href="#">191</a> , <a href="#">191</a> , <a href="#">192</a> , <a href="#">193</a>	Added “Delivery State” description to UCB_BMHDX_*, UCB_PFLASH_*, UCB_DFLASH_*, UCB_DBG_*, UCB_HSM_*, UCB_HSMCOTP0/1_*, UCB_ECPRIOD_*, UCB_SWAP_*, UCB_OTPy_*, UCB_REDSEC.	
<a href="#">Page 197</a>	UCB_SWAP: Changed in description of MARKERHx from “... corresponding MARKERL0.SWAP ...” to “... corresponding MARKERLx.SWAP ...”. Same change done in description of its bitfield MARKERHx.ADDR.	
<a href="#">Page 198</a>	UCB_SWAP: Changed in description of CONFIRMATIONHx from “... corresponding CONFIRMATIONL0.CODE ...” to “... corresponding CONFIRMATIONLx.CODE ...”. Same change done in description of its bitfield CONFIRMATIONHx.CODE.	
<b>V2.0.21</b>		
-	Only changes to IFX internal content.	
<b>V2.0.22</b>		
<a href="#">Page 182</a>	In Table “On Chip Bus Address Map of User Configuration Blocks” corrected name of UCB_OPT0_COPY to UCB_OTP0_COPY.	

## Local Memory Unit (LMU)

# 7 Local Memory Unit (LMU)

The Local Memory Unit is an SRI peripheral providing access to volatile memory resources. Its primary purpose is to provide up to 256 KiB of local memory for general purpose usage. A product may contain multiple instances of the LMU. Refer to the system memory map for the configuration applicable to each product. Each instance of the LMU has its own set of configuration registers.

Data stored in the local memory is protected by ECC at all points within the LMU. Areas of local memory can be write protected by configuring up to sixteen address ranges using SFRs in the LMU. Each of these ranges can be sized in thirty-two byte increments and has its own, independent list of Master Tag IDs permitted write access. Read accesses are not protected.

## 7.1 Feature List

An overview of the features implemented in the LMU follows:

- Up to 256 KiB of SRAM
  - organized as 64 bit words
  - support for byte, half word and word accesses as well as double-word and burst accesses
  - memory can be used as overlay memory
- Protection of LMU SRAM contents
  - sixteen programmable address regions can be protected
  - each address range has a programmable list of bus masters permitted read or write access based on the Unique master Tag ID

## 7.2 Functional Description

### 7.2.1 Local Memory (LMU SRAM)

The LMU SRAM can be used for code execution, data storage or overlay memory. The address range of the memory is defined in the system memory map. As well as being accessed via cached (segment 9<sub>H</sub>), the memory can be accessed via non-cached (segment B<sub>H</sub>) memory addresses.

The memory implements memory integrity checking for error detection and correction. This means that the memory must be initialized before reads are attempted with the integrity checking enabled to avoid generating spurious data corruption errors. Initializing with the memory integrity logic disabled allows the LMU SRAM to support initialisation using word (32 bit) or smaller writes as well as 64 bit writes.

If memory integrity checking is enabled, a read access which fails the integrity check will cause an error condition to be flagged to the initiated bus master. This behaviour can be changed by setting the MEMCON.ERRDIS bit to 1<sub>B</sub>. If ERRDIS is set, the access will terminate normally.

An ECC error will also be reported to the SMU. The SMU will use this signal for error indication and triggering of an NMI trap (if enabled).

The LMU SRAM is internally organized as a 64 bit memory without the possibility of sub-word accesses. This means that any write access of less than 64 bits of data needs an internal Read-Modify-Write (iRMW) operation to correctly write the data and update the ECC data. This happens transparently to rest of the system unless an uncorrected ECC error is detected during the read phase of the iRMW. This ECC error will be flagged to the SMU in the same way as a data ECC error occurring during a normal read. In addition the MEMCON.RMWERR flag will be set. The write operation will not take place as this would write incorrect ECC data to the memory (the ECC would match the written data but the data would potentially contain an uncorrected error).

## Local Memory Unit (LMU)

LMU SRAM performance will be the same as, or better than, the performance of the embedded flash. This applies to both the initial latency of the first word returned and also the incremental latency for each word in the same cache line fetched.

If the CPU access cannot be handled by the LMU SRAM (e.g. an attempt has been made to access RAM with the LMU clock disabled), an SRI bus error is reported by the LMU. This will, for example, cause a CPU to take a DSE trap if the access is from a CPU and a DMA access to terminate with an error condition.

The ERRDIS bitfield of the MEMCON register is protected by MEMCON.PMIC bit. If the data written to the register has the bitfield set to 0<sub>B</sub>, no change will be made to ERRDIS (bit 9<sub>D</sub> of the register) regardless of the data written to the field.

### 7.2.2 Memory Protection

The LMU allows for the definition of sixteen protected regions of SRAM memory. The protection applies only to accesses to SRAM included in the LMU, not registers.

The protection scheme is based on the use of Unique Master Tag IDs to identify the master attempting the access and allows for a six bit tag individually identifying up to 64 masters.

Each region is defined using six registers:

- RGNLAX (x=0-15) to define the lower address of the region, RGNUAx (x=0-15) to define the upper address of the region
- two registers RGNACCENWAX (x=0-15) and RGNACCENWBx (x=0-15), to individually select the master tags permitted write access to the defined address range
- two registers RGNACCENRAx (x=0-15) and RGNACCENRBx (x=0-15) to individually select the master tags permitted read access to the defined address range.

The scheme is compatible with the Tricore implementation so RGNLAX (x=0-15) defines the first address in the region.

After reset, the region address registers will be set to include the whole of the LMU SRAM address space and read and write access by all masters will be enabled.

The registers implementing the memory protection scheme are protected by the “safety endinit” function.

If overlapping regions are defined, then an access only needs to be permitted by one of the overlapping regions for it to succeed.

When altering protection settings, it should be noted that, due to access pipelining in the LMU and resynchronization delays in the register block, an access to a memory address affected by the protection change occurring immediately after the register write initiating the change may, or may not, be affected by the changed settings.

For the Cerberus and HSM masters, if present in the product, access protection only applies to write accesses. Read accesses from these masters are always successful.

### 7.2.3 LMU Register Protection

The LMU implements the standard register protection scheme for peripheral registers using the ACCEN0 and ACCEN1 registers. This allows the LMU control registers to be protected from write accesses by untrusted masters. Masters are identified using the SRI tag of the access and, if the appropriate bit is not set in the access enable registers, write accesses will be disconnected with error acknowledge. See the On Chip Bus chapter for the product’s master Tag ID to master peripheral mapping. This protection scheme does not apply to the ACCEN0 and ACCEN1 themselves.

ACCEN0, ACCEN1 and SMCTRL are protected by Safe Endinit while all other registers are Endinit protected. The Endinit and Safe Endinit system status is defined by different Watchdog Units in the System Control Unit (SCU).

## Local Memory Unit (LMU)

### 7.2.4 Error Detection and Signalling

The LMU will detect several different classes of error which cannot necessarily be signalled on the SRI during the associated transaction. However, all errors will cause a trigger to be sent to the SMU for processing. Some will additionally cause a flag to be set in the MEMCON register. Unless explicitly stated below, the access will complete. The following list details the detected error conditions:

#### 7.2.4.1 SRI access address phase error

If an ECC error occurs on the address phase of an SRI access then the MEMCON.ADDERR bit will be set and an error will be signalled to the SMU. The SRI access will abort without attempting to modify RAM or SFR contents.

#### 7.2.4.2 SRI write access data phase error

If an ECC error occurs on the data phase of an SRI write access then the MEMCON.DATAERR bit will be set and an error will be signalled to the SMU. The write completes without signalling an SRI error.

#### 7.2.4.3 Uncorrected ECC Error

If an uncorrected ECC error is reported by the RAM during an SRAM read then an error will be signalled to the SMU.

#### 7.2.4.4 SRAM Data Correction ECC failure

The hardware used to check and correct the read data from the SRAM is replicated and the output from the two instances is compared. In the event of a difference between the two outputs, an error condition will be signalled to the SMU. The second instance will use inverted logic to eliminate common failure modes.

#### 7.2.4.5 Internal Data Transfer ECC Error

Internal registers used to transfer data between the RAM and the SRI interface are ECC protected. In the event of this ECC detecting an error, the MEMCON.INTERR bit will be set and an error will be signalled to the SMU.

#### 7.2.4.6 Access Protection Violation

If either the memory protection or register protection detect a protection violation, then the violating access will

- be terminated with an error if a read
- fail silently if a write

In both cases, the error will be signalled to the SMU.

#### 7.2.4.7 Internal SRAM Read Error

The LMU will perform a internal Read-Modify-Write (iRMW) access when a write of less than 64 bits of data is performed. An ECC error reported by the RAM on the read phase will cause the MEMCON.RMWERR bit to be set and an error will be signalled to the SMU. The write phase of the iRMW will not take place unless the MEMCON.ERRDIS bit is set.

#### 7.2.4.8 Control Logic Failure

The control logic of the LMU which is not suitable for protection by one of the ECC checks will be lockstepped (duplicated and compared). Any mismatch between the two copies of the logic will be detected and an alarm generated. This mechanism will be enabled after reset and can be disabled by writing  $01_B$  (OFF) to the SCTRL.LSEN bitfield. The current status of the lockstep is reported by SCTRL.LSSTAT. The alarm signal can be

## Local Memory Unit (LMU)

tested by writing  $10_B$  to SCTRL.LSTST. The lockstep block also runs a background self test which periodically checks the comparator functionality. A self test failure will trigger an SMU alarm.

The lockstep block also checks consistency of its own control state by maintaining all such information in redundant pairs of flip-flops where one flip-flop is the logical inverse of the other. Any of these pairs being in an inconsistent state (i.e.  $00_B$  or  $11_B$ ) will trigger a separate SMU alarm.

### 7.2.5 SRAM Data Correction ECC failure

The hardware used to check and correct the read data from the SRAM is replicated and the data output from the two instances is compared. In the event of a difference between the two data outputs, an error condition will be signalled to the SMU. The second instance will use inverted logic to eliminate common failure modes.

### 7.2.6 Internal Data Transfer ECC Error

Internal registers used to transfer data between the RAM and the SRI interface are ECC protected. In the event of this ECC detecting an error, the MEMCON.INTERR bit will be set and an error will be signalled to the SMU.

### 7.2.7 Internal SRAM Read Error

The LMU will perform a internal Read-Modify-Write (iRMW) access when a write of less than 64 bits of data is performed. An ECC error reported by the RAM on the read phase will cause the MEMCON.RMWERR bit to be set and an error will be signalled to the SMU. The write phase of the iRMW will not take place unless the MEMCON.ERRDIS bit is set.

### 7.2.8 Clock Control

The LMU contains a clock control register, CLC, which allows the LMU to be put into a power saving mode. If LMU \_CLC.DISR is set then the LMU will be disabled and all accesses will be errored unless they are addressed to a register.

## Local Memory Unit (LMU)

### 7.3 LMU Registers

The registers of each LMU instance are mapped into a 64 kByte address space. Accesses to unused register space will cause an SRI bus error.

**Table 239 Register Overview - Common (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
CLC	LMU Clock Control Register	00000 <sub>H</sub>	SV	SV,E,P	Application Reset	<a href="#">Page 5</a>
MODID	LMU Module ID Register	00008 <sub>H</sub>	SV	BE	Application Reset	<a href="#">Page 6</a>
ACCENO	LMU Access Enable Register 0	00010 <sub>H</sub>	SV	SV,SE	Application Reset	<a href="#">Page 6</a>
ACCEN1	LMU Access Enable Register 1	00014 <sub>H</sub>	SV	SV,SE	Application Reset	<a href="#">Page 7</a>
MEMCON	LMU Memory Control Register	00020 <sub>H</sub>	SV	SV,E,P	Application Reset	<a href="#">Page 7</a>
SCTRL	LMU Safety Control Register	00024 <sub>H</sub>	SV	SV,SE,P	Application Reset	<a href="#">Page 8</a>
RGNLAX	LMU Region Lower Address Register	00050 <sub>H</sub> + x*10 <sub>H</sub>	SV	SV,SE,P	Application Reset	<a href="#">Page 10</a>
RGNUAX	LMU Region Upper Address Register	00054 <sub>H</sub> + x*10 <sub>H</sub>	SV	SV,SE,P	Application Reset	<a href="#">Page 10</a>
RGNACCENWAX	LMU Region Write Access Enable Register A	00058 <sub>H</sub> + x*10 <sub>H</sub>	SV	SV,SE,P	Application Reset	<a href="#">Page 11</a>
RGNACCENWBX	LMU Region Write Access Enable Register B	0005C <sub>H</sub> + x*10 <sub>H</sub>	SV	SV,SE,P	Application Reset	<a href="#">Page 12</a>
RGNACCENRAx	LMU Region Read Access Enable Register A	00158 <sub>H</sub> + x*10 <sub>H</sub>	SV	SV,SE,P	Application Reset	<a href="#">Page 12</a>
RGNACCENRBx	LMU Region Read Access Enable Register B	0015C <sub>H</sub> + x*10 <sub>H</sub>	SV	SV,SE,P	Application Reset	<a href="#">Page 13</a>

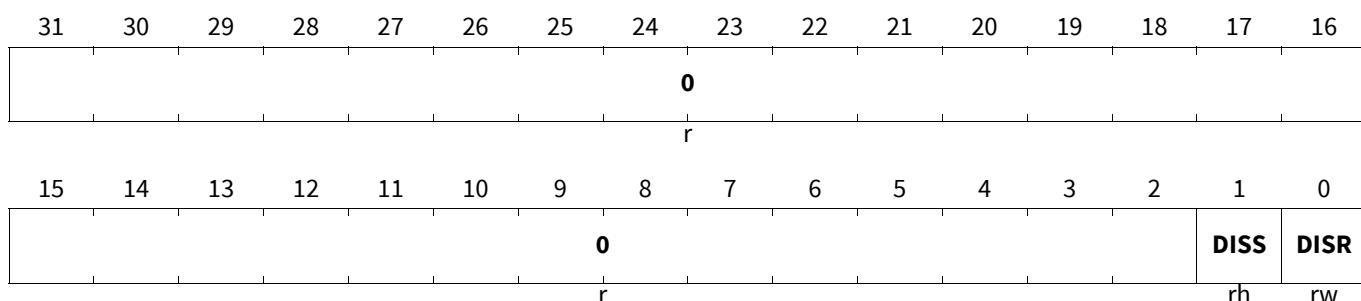
#### LMU Clock Control Register

##### CLC

##### LMU Clock Control Register

(00000<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



## Local Memory Unit (LMU)

Field	Bits	Type	Description
<b>DISR</b>	0	rw	<b>LMU_instance_nameDisable Request Bit</b> This bit is used for enable/disable control of the LMU. $0_B$ LMU disable is not requested $1_B$ LMU disable is requested
<b>DISS</b>	1	rh	<b>LMU_instance_nameDisable Status Bit</b> Current state of LMU. $0_B$ LMU is enabled (default after reset) $1_B$ LMU is disabled
<b>0</b>	31:2	r	<b>Reserved - RES</b>

### LMU Module ID Register

#### MODID

**LMU Module ID Register** **(00008<sub>H</sub>)** **Application Reset Value: 0088 C003<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ID_VALUE															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID_VALUE															

Field	Bits	Type	Description
<b>ID_VALUE</b>	31:0	r	<b>Module Identification Value</b>

### LMU Access Enable Register 0

The Access Enable Register 0 controls access for transactions to registers with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The LMU is prepared for an 6 bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ... ,EN31 -> TAG ID 011111B.

#### ACCEN0

**LMU Access Enable Register 0** **(00010<sub>H</sub>)** **Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN31</b>	<b>EN30</b>	<b>EN29</b>	<b>EN28</b>	<b>EN27</b>	<b>EN26</b>	<b>EN25</b>	<b>EN24</b>	<b>EN23</b>	<b>EN22</b>	<b>EN21</b>	<b>EN20</b>	<b>EN19</b>	<b>EN18</b>	<b>EN17</b>	<b>EN16</b>
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EN15</b>	<b>EN14</b>	<b>EN13</b>	<b>EN12</b>	<b>EN11</b>	<b>EN10</b>	<b>EN9</b>	<b>EN8</b>	<b>EN7</b>	<b>EN6</b>	<b>EN5</b>	<b>EN4</b>	<b>EN3</b>	<b>EN2</b>	<b>EN1</b>	<b>ENO</b>
rw															

## **Local Memory Unit (LMU)**

Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<p><b>Access Enable for Master TAG ID n</b></p> <p>This bit enables access to the LMU register addresses for transactions with the Master TAG ID n</p> <p>0<sub>B</sub> Write access will terminate without error but will not be executed.</p> <p>1<sub>B</sub> Write and read accesses will be executed</p>

LMU Access Enable Register 1

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

ACCEN1

LMU Access Enable Register 1

(00014<sub>H</sub>)

## **Application Reset Value: FFFF FFFF**

Field	Bits	Type	Description
<b>ENn (n=32-63)</b>	n-32	rw	<p><b>Access Enable for Master TAG ID n</b></p> <p>This bit enables access to the LMU register addresses for transactions with the Master TAG ID n</p> <p>0<sub>B</sub> Write access will terminate without error but will not be executed.</p> <p>1<sub>B</sub> Write and read accesses will be executed</p>

LMU Memory Control Register

This register provides software with the capability to monitor both the memory integrity error checking and the error signalling to the SMU.

MEMCON

LMU Memory Control Register

(00020..)

**Application Reset Value:** 0000 0000..

The diagram illustrates a 16-bit memory address bus. It consists of two parallel buses: a top bus and a bottom bus. The top bus has bits 31 through 17, followed by a central bit labeled 0, and then bits 16 through 1. The bottom bus has bits 15 through 1, followed by a central bit labeled 0. Between the two buses, there are several control signals: **ERRDI S**, **PMIC**, **ADDE RR**, **DATAE RR**, **RMWE RR**, and **INTER R**. The labels **r** and **rwh** are placed under the respective bus segments to indicate their function.

**Local Memory Unit (LMU)**

Field	Bits	Type	Description
<b>INTERR</b>	2	rwh	<b>Internal ECC Error</b> Flag set by hardware when the LMU detects an ECC error on the internal data path registers while accessing the RAM. $0_B$ No error has occurred $1_B$ An error has been observed during a RAM access.
<b>RMWERR</b>	4	rwh	<b>Internal Read Modify Write Error</b> Flag set by hardware when an uncorrected ECC error is reported by the RAM on the read phase of an internal RMW operation. $0_B$ No error has occurred $1_B$ An error has been observed during an iRMW operation.
<b>DATAERR</b>	6	rwh	<b>SRI Data Phase ECC Error</b> Flag set by hardware when the SRI interface detects an ECC error in the data phase of an incoming write transaction. This bit is cleared by writing $0_B$ but cannot be set by software. $0_B$ No error has occurred $1_B$ An ECC error has been observed on an SRI transaction addressed to the LMU
<b>ADDERR</b>	7	rwh	<b>SRI Address Phase ECC Error</b> Flag set by hardware when the SRI interface detects an ECC error in the address phase of an incoming transaction. This bit is cleared by writing $0_B$ but cannot be set by software. $0_B$ No error has occurred $1_B$ An ECC error has been observed on an SRI transaction addressed to the LMU
<b>PMIC</b>	8	rwh	<b>Protection Bit for Memory Integrity Control Bit</b> Will always return $0_B$ when read $0_B$ Bit Protection: Bit 9 remains unchanged after MEMCON write. $1_B$ Bit 9 will be updated by the current write to MEMCON
<b>ERRDIS</b>	9	rw	<b>ECC Error Disable</b> When set to $1_B$ SRI error reporting of ECC errors in data read from the SRAM will be disabled and an ECC error on the read phase of an iRMW will not cause the write phase to be aborted. $0_B$ Normal behavior. SRI error will occur on SRAM ECC errors. Default after reset $1_B$ Test or Initialisation Mode. SRI errors will not be generated on an SRAM ECC error. This does not affect the generation of alarms.
<b>0</b>	1:0, 3, 5, 31:10	r	<b>Reserved</b> Read as $0_H$ , must be written as $0_H$

**LMU Safety Control Register**

This register provides control of the user configurable safety mechanisms of the LMU. Writing one of the invalid values to any of the redundant control fields will cause an SMU alarm.

## Local Memory Unit (LMU)

## SCTRL

## LMU Safety Control Register

(00024<sub>H</sub>)Application Reset Value: 0002 0600<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														LSSTAT	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				LSTST		LSEN		0				GEC		GED	
r				rw		rw			r			w	w		

Field	Bits	Type	Description
GED	0	w	<b>Generate Error in ECC for Data Protection</b> The data paths between the SRAM and the LMU bus interface are protected by ECC logic. This bit is used to inject an error into the next write access so that the SMU alarm can be tested. Reading this bit always returns 0 <sub>B</sub> . Writing works as follows: 0 <sub>B</sub> No Effect 1 <sub>B</sub> Inject error into next LMU RAM Write Access.
GEC	1	w	<b>Generate Error in ECC for Error Correction</b> The data read from the SRAM is corrected by ECC logic. This ECC logic is duplicated so the functionality can be checked. This bit is used to inject an error into the next read access so that the SMU alarm can be tested. Reading this bit always returns 0 <sub>B</sub> . Writing works as follows: 0 <sub>B</sub> No Effect 1 <sub>B</sub> Inject error into next LMU RAM Read Access.
LSEN	9:8	rw	<b>Lockstep Enable</b> Control of Lockstep comparators checking the duplicated logic area for errors. 00 <sub>B</sub> <b>RES0</b> , Invalid 01 <sub>B</sub> <b>OFF</b> , Lockstep Off 10 <sub>B</sub> <b>ON</b> , Lockstep On 11 <sub>B</sub> <b>RES3</b> , Invalid
LSTST	11:10	rw	<b>Lockstep Test</b> Setting this bitfield will inject an error into the lockstep comparators checking the duplicated logic area. This will allow the correct operation of the SMU alarm to be verified. An error will continue to be injected until this field is reset. 00 <sub>B</sub> <b>RES0</b> , Invalid 01 <sub>B</sub> <b>OFF</b> , No error injected 10 <sub>B</sub> <b>ON</b> , Error injected 11 <sub>B</sub> <b>RES3</b> , Invalid

## Local Memory Unit (LMU)

Field	Bits	Type	Description
<b>LSSTAT</b>	17:16	rh	<b>Lockstep Status</b> Reports the status of the lockstep comparators. 00 <sub>B</sub> <b>RES0</b> , Invalid 01 <sub>B</sub> <b>OFF</b> , Lockstep is Off 10 <sub>B</sub> <b>ON</b> , Lockstep is On 11 <sub>B</sub> <b>RES3</b> , Invalid
<b>0</b>	7:2, 15:12, 31:18	r	<b>Reserved</b> Read as 0 <sub>H</sub> , must be written as 0 <sub>H</sub>

### LMU Region Lower Address Register

In conjunction with the associated RGNUA and RGNACCENx registers, the RGNLA register provides control of a memory protection region. The register is cleared by an application (Class 3) reset. RGNLA defines the lower address of a region of memory, RGNUA defines the upper address and the RGNACCENx registers define the Unique Master Tag IDs allowed to access the region. Address ranges can be set to be larger than the LMU address space but only accesses to the LMU are affected by these registers.

#### RGNL<sub>A</sub>x (x=0-15)

LMU Region Lower Address Register (00050 <sub>H</sub> +x*10 <sub>H</sub> ) Application Reset Value: 0000 0000 <sub>H</sub>																																																															
<table border="1"> <thead> <tr> <th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr> </thead> <tbody> <tr> <td align="center" colspan="16" style="text-align: center;">ADDR</td></tr> <tr> <td align="center" colspan="16" style="text-align: center;">rw</td></tr> </tbody> </table>																31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	ADDR																rw															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16																																																
ADDR																																																															
rw																																																															
<table border="1"> <thead> <tr> <th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr> </thead> <tbody> <tr> <td align="center" colspan="10" style="text-align: center;">ADDR</td><td align="center" colspan="6" style="text-align: center;">0</td></tr> <tr> <td align="center" colspan="10" style="text-align: center;">rw</td><td align="center" colspan="6" style="text-align: center;">r</td></tr> </tbody> </table>																15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	ADDR										0						rw										r					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																
ADDR										0																																																					
rw										r																																																					

Field	Bits	Type	Description
<b>ADDR</b>	31:5	rw	<b>Region Lower Address</b> Bits 31 to 5 of the SRI address which is the lower bound of the defined memory region
<b>0</b>	4:0	r	<b>Reserved</b> Read as 0 <sub>B</sub> , should be written with 0 <sub>B</sub> although written value will be ignored.

### LMU Region Upper Address Register

In conjunction with the associated RGNLA and RGNACCENx registers, the RGNUA register provides control of a memory protection region. The register is cleared by an application (Class 3) reset. RGNUA defines the upper address of the memory region and will contain the first address outside the protected region.

## Local Memory Unit (LMU)

### RGNUAx (x=0-15)

**LMU Region Upper Address Register (00054<sub>H</sub>+x\*10<sub>H</sub>) Application Reset Value: FFFF FFE0<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR															
r															

Field	Bits	Type	Description
<b>ADDR</b>	31:5	rw	<b>Region Lower Address</b> Bits 31 to 5 of the SRI address which is the upper bound of the defined memory region. i.e. the first address outside the protected region.
<b>0</b>	4:0	r	<b>Reserved</b> Read as 0 <sub>B</sub> , should be written with 0 <sub>B</sub> although written value will be ignored.

### LMU Region Write Access Enable Register A

The Write Access Enable Register A controls write access for transactions to the protected memory region with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The LMU supports a 6 bit TAG ID. The registers RGNACCENWAI / RGNACCENWBi provide one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to RGNACCENWAi.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ... ,EN31 -> TAG ID 011111B.

### RGNACCENWAx (x=0-15)

**LMU Region Write Access Enable Register A (00058<sub>H</sub>+x\*10<sub>H</sub>) Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN31</b>	<b>EN30</b>	<b>EN29</b>	<b>EN28</b>	<b>EN27</b>	<b>EN26</b>	<b>EN25</b>	<b>EN24</b>	<b>EN23</b>	<b>EN22</b>	<b>EN21</b>	<b>EN20</b>	<b>EN19</b>	<b>EN18</b>	<b>EN17</b>	<b>EN16</b>
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EN15</b>	<b>EN14</b>	<b>EN13</b>	<b>EN12</b>	<b>EN11</b>	<b>EN10</b>	<b>EN9</b>	<b>EN8</b>	<b>EN7</b>	<b>EN6</b>	<b>EN5</b>	<b>EN4</b>	<b>EN3</b>	<b>EN2</b>	<b>EN1</b>	<b>ENO</b>
rw															

## Local Memory Unit (LMU)

Field	Bits	Type	Description
ENn (n=0-31)	n	rw	<p><b>Access Enable for Master TAG ID n</b></p> <p>This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n</p> <p>0<sub>B</sub> <b>DISW</b>, No write accesses identified with tag n are permitted for this region . Write Accesses will terminate silently without modifying RAM contents.</p> <p>1<sub>B</sub> <b>ENW</b>, Write Accesses that are identified with tag n are permitted for this region</p>

### LMU Region Write Access Enable Register B

The Write Access Enable Register B controls write access for transactions to the protected memory region with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The LMU supports a 6 bit TAG ID. The registers RGNACCENWB<sub>i</sub> / RGNACCENWB<sub>i</sub> provide one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to RGNACCENWB<sub>i</sub>.ENx: EN0 -> TAG ID 100000<sub>B</sub>, EN1 -> TAG ID 100001<sub>B</sub>, ... ,EN31 -> TAG ID 111111<sub>B</sub>.

#### RGNACCENWB<sub>x</sub> (x=0-15)

#### LMU Region Write Access Enable Register B (0005C<sub>H</sub>+x\*10<sub>H</sub>)

Application Reset Value: FFFF FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw															

Field	Bits	Type	Description
ENn (n=32-63)	n-32	rw	<p><b>Access Enable for Master TAG ID n</b></p> <p>This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n</p> <p>0<sub>B</sub> <b>DISW</b>, No write accesses identified with tag n are permitted for this region . Write Accesses will terminate silently without modifying RAM contents.</p> <p>1<sub>B</sub> <b>ENW</b>, Write Accesses that are identified with tag n are permitted for this region</p>

### LMU Region Read Access Enable Register A

The Read Access Enable Register A controls read access for transactions to the protected memory region with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The LMU supports a 6 bit TAG ID. The registers RGNACCENRA<sub>i</sub> / RGNACCENRB<sub>i</sub> provide one enable bit for each possible 6 bit TAG ID encoding.

## Local Memory Unit (LMU)

Mapping of TAG IDs to RGNACCENRAi.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ... ,EN31 -> TAG ID 011111B. Note that the accesses from the Cerberus, HSM and IOC32 will always be permitted regardless of the value in the register.

### RGNACCENRAx (x=0-15)

#### LMU Region Read Access Enable Register A (00158<sub>H</sub>+x\*10<sub>H</sub>)

Application Reset Value: FFFF FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

Field	Bits	Type	Description
ENn (n=0-31)	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables read access to the module kernel addresses for transactions with the Master TAG ID n 0 <sub>B</sub> <b>DISR</b> , No read accesses identified with tag n are permitted for this region. Read accesses identified with tag n will terminate with an error condition 1 <sub>B</sub> <b>ENR</b> , Read accesses identified with tag n are permitted for this region

### LMU Region Read Access Enable Register B

The Read Access Enable Register B controls read access for transactions to the protected memory region with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The LMU supports a 6 bit TAG ID. The registers RGNACCENRAi / RGNACCENRBi provide one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to RGNACCENRBi.ENx: EN0 -> TAG ID 100000<sub>B</sub>, EN1 -> TAG ID 100001<sub>B</sub>, ... ,EN31 -> TAG ID 111111<sub>B</sub>. Note that the accesses from the Cerberus, HSM and IOC32 will always be permitted regardless of the value in the register.

### RGNACCENRBx (x=0-15)

#### LMU Region Read Access Enable Register B (0015C<sub>H</sub>+x\*10<sub>H</sub>)

Application Reset Value: FFFF FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw															

## Local Memory Unit (LMU)

Field	Bits	Type	Description
<b>ENn (n=32-63)</b>	n-32	rw	<p><b>Access Enable for Master TAG ID n</b></p> <p>This bit enables read access to the module kernel addresses for transactions with the Master TAG ID n</p> <p>0<sub>B</sub> <b>DISR</b>, No read accesses identified with tag n are permitted for this region. Read accesses identified with tag n will terminate with an error condition</p> <p>1<sub>B</sub> <b>ENR</b>, Read accesses identified with tag n are permitted for this region</p>

## 7.4 IO Interfaces

**Table 240 List of LMU Interface Signals**

Interface Signals	I/O	Description
SX_SRI		sri slave interface
		sri slave interface (RAM Address Range non-cached)
		sri slave interface (RAM Address Range cached)
SX_ALARM_LMU		LMU Alarm Outputs to SMU

---

**Local Memory Unit (LMU)****7.5 Revision History****Table 241 Revision History**

Reference	Change to Previous Version	Comment
V3.1.15	Revision history update, no functional changes.	
V3.1.16	- No functional change.	

## Default Application Memory (DAM)

### 8 Default Application Memory (DAM)

The Default Application Memory is an SRI peripheral providing access to volatile memory resources. Its primary purpose is to provide 64 kBytes or 32 kBytes of local memory for general purpose usage. The amount of memory available depends on the product.

Data stored in the local memory is protected by ECC. Areas of local memory can be write protected by configuring up to eight address ranges using SFRs in the DAM. Each of these ranges can be sized in thirty-two byte increments and has its own, independent list of Master Tag IDs permitted write access.

#### 8.1 Feature List

An overview of the features implemented in the DAM follows:

- 64 KiB of SRAM depending on the product
  - organized as 64 bit words
  - support for byte, half word and word accesses as well as double-word and burst accesses
- Protection of DAM SRAM contents
  - eight programmable address regions can be protected
  - each address range has a programmable list of bus masters permitted read or write access based on the Unique master Tag ID

#### 8.2 Functional Description

##### 8.2.1 Local Memory (DAM SRAM)

The DAM SRAM can be used for code execution or data storage but it does not have hardware safety mechanisms in place to support use in ASIL B, ASIL C or ASIL D applications. The address range of the memory is defined in the system memory map. As well as being accessed via cached (segment 9<sub>H</sub>), the memory can be accessed via non-cached (segment B<sub>H</sub>) memory addresses.

The SRAM can also be configured to emulate ROM by setting the ROM bit in the MEMCON register. In this mode, all write accesses from the SRI are terminated with an SRI error without changing the memory contents.

The memory implements memory integrity checking for error detection and correction. This means that the memory must be initialized before reads are attempted with the integrity checking enabled to avoid generating spurious data corruption errors. Initializing with the memory integrity logic disabled allows the DAM SRAM to support initialisation using word (32 bit) or smaller writes as well as 64 bit writes.

If memory integrity checking is enabled, a read access which fails the integrity check will be terminated with an SRI error condition. This behavior can be changed by setting the MEMCON.ERRDIS bit to 1<sub>B</sub>. If the bit is set then an SRI error will not occur.

An ECC error will also be reported to the SMU. The SMU will use this signal for error indication and triggering of an NMI trap (if enabled).

The DAM SRAM is internally organized as a 64 bit memory without the possibility of sub-word accesses. This means that any write access of less than 64 bits of data needs an internal Read-Modify-Write (iRMW) operation to correctly write the data and update the ECC data. This happens transparently to rest of the system unless an uncorrected ECC error is detected during the read phase of the iRMW. This ECC error will be flagged to the SMU in the same way as a data ECC error occurring during a normal read. In addition the DAM MEMCON.RMWERR flag will be set. The write operation will not take place as this would write incorrect ECC data to the memory (the ECC would match the written data but the data would potentially contain an uncorrected error).

## **Default Application Memory (DAM)**

If the access cannot be handled by the DAM SRAM (e.g. an attempt has been made to access RAM with the DAM clock disabled), an SRI bus error is reported by the DAM.

The ERRDIS bitfield of the MEMCON register is protected by MEMCON.PMIC bit. If the data written to the register has the bitfield set to  $0_B$ , no change will be made to ERRDIS (bit  $9_D$  of the register) regardless of the data written to the field.

### **8.2.2 Memory Protection**

The DAM allows for the definition of eight protected regions of SRAM memory. The protection applies only to accesses to SRAM included in the DAM, not registers.

The protection scheme is based on the use of Unique Master Tag IDs to identify the master attempting the access and allows for a six bit tag individually identifying up to 64 masters.

Each region is defined using six registers:

- RGNL<sub>A</sub>x (x=0-7) to define the lower address of the region, RGNU<sub>A</sub>x (x=0-7) to define the upper address of the region
- two registers RGNACCENWA<sub>x</sub> (x=0-7) and RGNACCENWB<sub>x</sub> (x=0-7), to individually select the master tags permitted write access to the defined address range
- two registers RGNACCENRA<sub>x</sub> (x=0-7) and RGNACCENRB<sub>x</sub> (x=0-7) to individually select the master tags permitted read access to the defined address range.

The scheme is compatible with the Tricore implementation so RGNL<sub>A</sub>x (x=0-7) defines the first address in the region.

After reset, the region address registers will be set to include the whole of the DAM SRAM address space and access by all masters will be enabled.

The registers implementing the memory protection scheme are protected by the “safety endinit” function.

If overlapping regions are defined, then an access only needs to be permitted by one of the overlapping regions for it to succeed.

When altering protection settings, it should be noted that, due to access pipelining in the DAM and resynchronization delays in the register block, an access to a memory address affected by the protection change occurring immediately after the register write initiating the change may, or may not, be affected by the changed settings.

For the Cerberus and HSM masters, if present in the product, access protection only applies to write accesses. Read accesses from these masters are always successful. Consequently the relevant bits in the RGNACCENRA<sub>x</sub> (x=0-7) or RGNACCENRB<sub>x</sub> (x=0-7) registers will always read as  $1_B$ .

### **8.2.3 DAM Register Protection**

The DAM implements the standard register protection scheme for peripheral registers using the ACCEN0 and ACCEN1 registers. This allows the DAM control registers to be protected from write accesses by untrusted masters. Masters are identified using the SRI tag of the access and, if the appropriate bit is not set in the access enable registers, write accesses will be disconnected with error acknowledge. See the On Chip Bus chapter for the product’s master Tag ID to master peripheral mapping. This protection scheme does not apply to the ACCEN0 and ACCEN1 themselves.

ACCEN0 and ACCEN1 are protected by Safe Endinit while all other registers are Endinit protected. The Endinit and Safe Endinit system status is defined by different Watchdog Units in the System Control Unit (SCU).

---

## Default Application Memory (DAM)

### 8.2.4 Error Detection and Signalling

The DAM will detect several different classes of error which cannot be signalled on the SRI during the associated transaction. These will cause a flag to be set in the MEMCON register and a trigger will be sent to either the SMU or the Interrupt system for processing. Unless explicitly stated below, the access will complete. The following list details the detected error conditions:

#### 8.2.4.1 SRI access address phase error

If an ECC error occurs on the address phase of an SRI access then the MEMCON.ADDERR bit will be set and an error will be signalled to the SMU. The SRI access will terminate with an error.

#### 8.2.4.2 SRI write access data phase error

If an ECC error occurs on the data phase of an SRI write access then the MEMCON.DATAERR bit will be set and an error will be signalled to the SMU

#### 8.2.4.3 Uncorrected ECC Error

If an uncorrected ECC error is signalled during an SRAM read then an error will be signalled to the SMU.

#### 8.2.4.4 Access Protection Violation

If either the memory protection or register protection detect a protection violation, then the violating access will

- be terminated with an error if a read
- be terminated with an error if a write

In both cases, the error signalled to the SMU.

### 8.2.5 Clock Control

The DAM contains a clock control register, CLC, which allows the DAM to be put into a power saving mode.

If CLC.DISR is set then the DAM will be disabled and all accesses will be errored unless they are addressed to a register. Accesses to DAM memory should not be attempted once the DAM has been disabled as accesses running when the register bit sets can either be cancelled or completed.

## Default Application Memory (DAM)

### 8.3 Registers

The DAM registers are mapped into a 32 kByte address space. Accesses to unused register space will cause an SRI bus error.

All registers are endinit or safe endinit protected and are accessible in Supervisor mode only.

**Table 242 Register Overview - SFRs (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
CLC	DAM Clock Control Register	00000 <sub>H</sub>	SV	SV,E,P	Application Reset	4
MODID	DAM Module ID Register	00008 <sub>H</sub>	SV	BE	Application Reset	5
ACCEN0	DAM Access Enable Register 0	00010 <sub>H</sub>	SV	SV,SE	Application Reset	5
ACCEN1	DAM Access Enable Register 1	00014 <sub>H</sub>	SV	SV,SE	Application Reset	6
MEMCON	DAM Memory Control Register	00020 <sub>H</sub>	SV	SV,E,P	Application Reset	6
RGNLAX	DAM Region Lower Address Register	00050 <sub>H</sub> + x*10 <sub>H</sub>	SV	SV,SE,P	Application Reset	8
RGNUAX	DAM Region Upper Address Register	00054 <sub>H</sub> + x*10 <sub>H</sub>	SV	SV,SE,P	Application Reset	8
RGNACCENWAX	DAM Region Write Enable Register A	00058 <sub>H</sub> + x*10 <sub>H</sub>	SV	SV,SE,P	Application Reset	9
RGNACCENWBX	DAM Region Write Enable Register B	0005C <sub>H</sub> + x*10 <sub>H</sub>	SV	SV,SE,P	Application Reset	10
RGNACCENRAx	DAM Region Read Enable Register A	000D8 <sub>H</sub> + x*10 <sub>H</sub>	SV	SV,SE,P	Application Reset	10
RGNACCENRBx	DAM Region Read Enable Register B	000DC <sub>H</sub> + x*10 <sub>H</sub>	SV	SV,SE,P	Application Reset	11

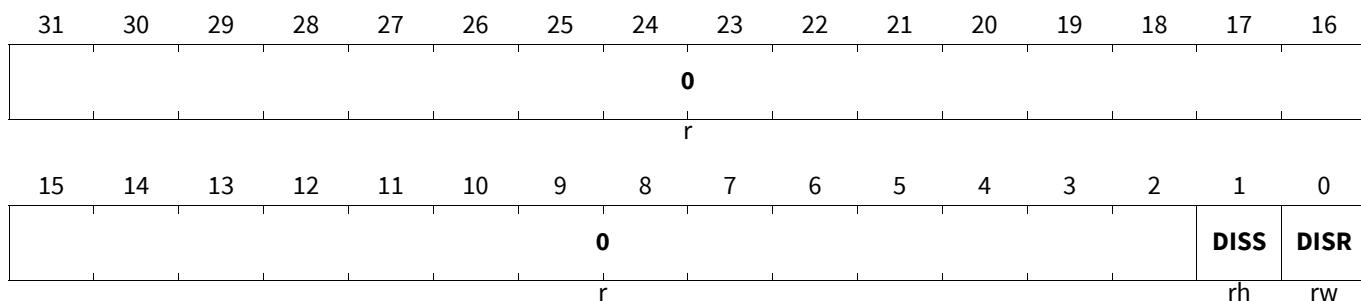
#### DAM Clock Control Register

##### CLC

##### DAM Clock Control Register

(00000<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



## Default Application Memory (DAM)

Field	Bits	Type	Description
<b>DISR</b>	0	rw	<b>DAM Disable Request Bit</b> This bit is used for enable/disable control of the DAM. 0 <sub>B</sub> DAM disable is not requested 1 <sub>B</sub> DAM disable is requested
<b>DISS</b>	1	rh	<b>DAM Disable Status Bit</b> Current state of DAM. 0 <sub>B</sub> DAM is enabled (default after reset) 1 <sub>B</sub> DAM is disabled
<b>0</b>	31:2	r	<b>Reserved - RES</b> Read as 0, must be written as 0

### DAM Module ID Register

#### MODID

<b>DAM Module ID Register</b>																<b>(00008<sub>H</sub>)</b>				<b>Application Reset Value: 0088 C003<sub>H</sub></b>																					
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16																										
<b>ID_VALUE</b>																																									
r																																									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	<b>ID_VALUE</b>																									
r																																									

Field	Bits	Type	Description
<b>ID_VALUE</b>	31:0	r	<b>Module Identification Value</b>

### DAM Access Enable Register 0

The Access Enable Register 0 controls write access for transactions to registers with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The DAM is prepared for an 6 bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6 bit TAG ID encoding.

<b>DAM Access Enable Register 0</b>																<b>(00010<sub>H</sub>)</b>				<b>Application Reset Value: FFFF FFFF<sub>H</sub></b>																			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16																								
<b>EN31</b>	<b>EN30</b>	<b>EN29</b>	<b>EN28</b>	<b>EN27</b>	<b>EN26</b>	<b>EN25</b>	<b>EN24</b>	<b>EN23</b>	<b>EN22</b>	<b>EN21</b>	<b>EN20</b>	<b>EN19</b>	<b>EN18</b>	<b>EN17</b>	<b>EN16</b>																								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw																								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																								
<b>EN15</b>	<b>EN14</b>	<b>EN13</b>	<b>EN12</b>	<b>EN11</b>	<b>EN10</b>	<b>EN9</b>	<b>EN8</b>	<b>EN7</b>	<b>EN6</b>	<b>EN5</b>	<b>EN4</b>	<b>EN3</b>	<b>EN2</b>	<b>EN1</b>	<b>ENO</b>																								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw																								

## Default Application Memory (DAM)

Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables access to the DAM register addresses for transactions with the Master TAG ID n 0 <sub>B</sub> Write access will terminate with error and will not be executed. 1 <sub>B</sub> Write and read accesses will be executed

### DAM Access Enable Register 1

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping).

#### ACCEN1

#### DAM Access Enable Register 1 (00014<sub>H</sub>) Application Reset Value: FFFF FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN63</b>	<b>EN62</b>	<b>EN61</b>	<b>EN60</b>	<b>EN59</b>	<b>EN58</b>	<b>EN57</b>	<b>EN56</b>	<b>EN55</b>	<b>EN54</b>	<b>EN53</b>	<b>EN52</b>	<b>EN51</b>	<b>EN50</b>	<b>EN49</b>	<b>EN48</b>
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EN47</b>	<b>EN46</b>	<b>EN45</b>	<b>EN44</b>	<b>EN43</b>	<b>EN42</b>	<b>EN41</b>	<b>EN40</b>	<b>EN39</b>	<b>EN38</b>	<b>EN37</b>	<b>EN36</b>	<b>EN35</b>	<b>EN34</b>	<b>EN33</b>	<b>EN32</b>
rw															

Field	Bits	Type	Description
<b>ENn (n=32-63)</b>	n-32	rw	<b>Access Enable for Master TAG ID n</b> This bit enables access to the DAM register addresses for transactions with the Master TAG ID n 0 <sub>B</sub> Write access will terminate with error and will not be executed. 1 <sub>B</sub> Write and read accesses will be executed

### DAM Memory Control Register

This register provides software with the capability to monitor both the memory integrity error checking and the error signalling to the SMU.

#### MEMCON

#### DAM Memory Control Register (00020<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
0																	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0								<b>ERRDI S</b>	<b>PMIC</b>	<b>ADDE RR</b>	<b>DATAE RR</b>	<b>0</b>	<b>RMWE RR</b>	<b>0</b>	<b>INTER R</b>	<b>0</b>	<b>ROM</b>
r								rw	w	rwh	rwh	r	rwh	r	rwh	r	rw

## Default Application Memory (DAM)

Field	Bits	Type	Description
<b>ROM</b>	0	rw	<b>Read Only Memory</b> Configure RAM to be Read Only Memory 0 <sub>B</sub> RAM can be written to from the SRI 1 <sub>B</sub> RAM cannot be written to from the SRI
<b>INTERR</b>	2	rwh	<b>Internal ECC Error</b> Flag set by hardware when the DAM logic detects an ECC error while accessing the RAM. This bit is cleared by writing 0 <sub>B</sub> but cannot be set by software. 0 <sub>B</sub> No error has occurred 1 <sub>B</sub> An error has been observed during a RAM access.
<b>RMWERR</b>	4	rwh	<b>Internal Read Modify Write Error</b> Flag set by hardware when the DAM logic detects an ECC error on the read phase of an internal RMW operation. This bit is cleared by writing 0 <sub>B</sub> but cannot be set by software. 0 <sub>B</sub> No error has occurred 1 <sub>B</sub> An error has been observed during an iRMW operation.
<b>DATAERR</b>	6	rwh	<b>SRI Data Phase ECC Error</b> Flag set by hardware when the SRI interface detects an ECC error in the data phase of an incoming write transaction. This bit is cleared by writing 0 <sub>B</sub> but cannot be set by software. 0 <sub>B</sub> No error has occurred 1 <sub>B</sub> An ECC error has been observed on an SRI transaction addressed to the DAM
<b>ADDERR</b>	7	rwh	<b>SRI Address Phase ECC Error</b> Flag set by hardware when the SRI interface detects an ECC error in the address phase of an incoming transaction. This bit is cleared by writing 0 <sub>B</sub> but cannot be set by software. 0 <sub>B</sub> No error has occurred 1 <sub>B</sub> An ECC error has been observed on an SRI transaction addressed to the DAM
<b>PMIC</b>	8	w	<b>Protection Bit for Memory Integrity Control Bit</b> Will always return 0 <sub>B</sub> when read 0 <sub>B</sub> Bit Protection: Bit 9 remains unchanged after MEMCON write. 1 <sub>B</sub> Bit 9 will be updated by the current write to MEMCON
<b>ERRDIS</b>	9	rw	<b>ECC Error Disable</b> When set SRI bus errors caused by ECC errors in data read from the SRAM will be disabled 0 <sub>B</sub> Normal behavior. SRI error will occur on SRAM ECC errors. Default after reset 1 <sub>B</sub> Test Mode. SRI errors will not be generated on an SRAM ECC error. This does not affect the generation of interrupts.

## Default Application Memory (DAM)

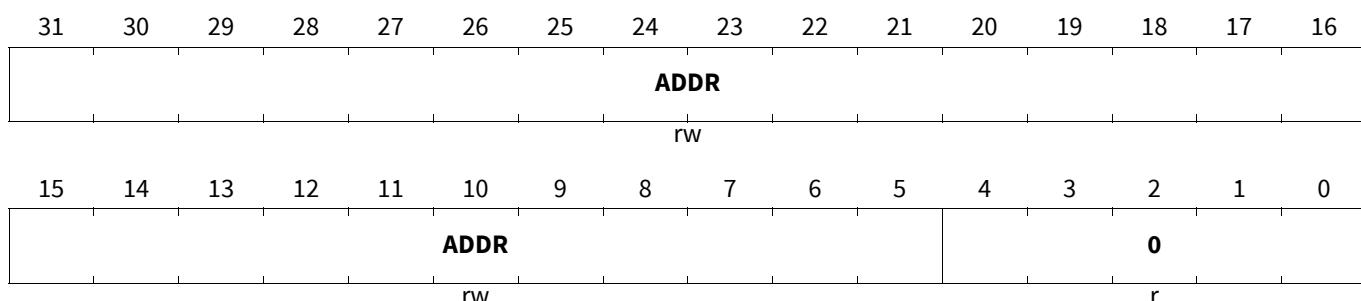
Field	Bits	Type	Description
0	1, 3, 5, 15:10, 31:16	r	<b>Reserved</b> Read as 0, must be written as 0

### DAM Region Lower Address Register

In conjunction with the associated RGNUA and RGNACCEN\*x registers, the RGNLA register provides control of a memory protection region. The register is cleared by a application (Class 3) reset. RGNLA defines the lower address of a region of memory, RGNUA defines the upper address and the RGNACCEN\*x registers define the Unique Master Tag IDs allowed to access the region. Address ranges can be set to be larger than the DAM address space but only accesses to the DAM are affected by these registers.

#### RGNLAX (x=0-7)

**DAM Region Lower Address Register** **(00050<sub>H</sub>+x\*10<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>ADDR</b>	31:5	rw	<b>Region Lower Address</b> Bits 31 to 5 of the SRI address which is the lower bound of the defined memory region
0	4:0	r	<b>Reserved</b> Read as 0 <sub>B</sub> , should be written with 0 <sub>B</sub> although written value will be ignored.

### DAM Region Upper Address Register

In conjunction with the associated RGNLA and RGNACCEN\*x registers, the RGNUA register provides control of a memory protection region. The register is cleared by an application (Class 3) reset. RGNUA defines the upper address of the memory region and will contain the first address outside the protected region.

## Default Application Memory (DAM)

### RGNUAx (x=0-7)

#### DAM Region Upper Address Register (00054<sub>H</sub>+x\*10<sub>H</sub>)

Application Reset Value: FFFF FFE0<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR															
r															

Field	Bits	Type	Description
ADDR	31:5	rw	<b>Region Lower Address</b> Bits 31 to 5 of the SRI address which is the upper bound of the defined memory region. i.e. the first address outside the protected region.
0	4:0	r	<b>Reserved</b> Read as 0 <sub>B</sub> , should be written with 0 <sub>B</sub> although written value will be ignored.

### DAM Region Write Enable Register A

The Write Access Enable Register A controls write access for transactions to the protected memory region with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The DAM is prepared for an 6 bit TAG ID. The registers RGNACCENWAI / RGNACCENWBi are providing one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to RGNACCENWAI.ENn: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ... ,EN31 -> TAG ID 011111B.

### RGNACCENWAX (x=0-7)

#### DAM Region Write Enable Register A

(00058<sub>H</sub>+x\*10<sub>H</sub>)

Application Reset Value: FFFF FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

Field	Bits	Type	Description
ENn (n=0-31)	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 <sub>B</sub> No write accesses identified with TAG n are permitted for this region. Writes accesses will terminate with an error condition. 1 <sub>B</sub> Write permitted for this region

## Default Application Memory (DAM)

### DAM Region Write Enable Register B

The Write Access Enable Register B controls write access for transactions to the protected memory region with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The DAM is prepared for an 6 bit TAG ID. The registers RGNACCENWB<sub>i</sub> / RGNACCENWB<sub>j</sub> are providing one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to RGNACCENWB<sub>i</sub>.EN<sub>n</sub>: EN0 -> TAG ID 100000<sub>B</sub>, EN1 -> TAG ID 100001<sub>B</sub>, ... ,EN31 -> TAG ID 111111<sub>B</sub>.

#### RGNACCENWB<sub>x</sub> (x=0-7)

**DAM Region Write Enable Register B** (0005C<sub>H</sub>+x\*10<sub>H</sub>) **Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw															

Field	Bits	Type	Description
EN <sub>n</sub> (n=32-63)	n-32	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 <sub>B</sub> No write accesses identified with TAG n are permitted for this region. Writes accesses will terminate with an error condition. 1 <sub>B</sub> Write permitted for this region

### DAM Region Read Enable Register A

The Read Access Enable Register A controls read access for transactions to the protected memory region with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The DAM is prepared for an 6 bit TAG ID. The registers RGNACCENRA<sub>i</sub> / RGNACCENRB<sub>j</sub> are providing one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to RGNACCENRA<sub>i</sub>.EN<sub>n</sub>: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ... ,EN31 -> TAG ID 011111B. Note that the accesses from the Cerberus, HSM and IOC32 will always be permitted regardless of the value in the register.

#### RGNACCENRA<sub>x</sub> (x=0-7)

**DAM Region Read Enable Register A** (000D8<sub>H</sub>+x\*10<sub>H</sub>) **Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rw															

## Default Application Memory (DAM)

Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<p><b>Access Enable for Master TAG ID n</b></p> <p>This bit enables read access to the module kernel addresses for transactions with the Master TAG ID n</p> <p><math>0_B</math> No read accesses are permitted for this region. Read accesses will terminate with an error condition</p> <p><math>1_B</math> Read permitted for this region</p>

### DAM Region Read Enable Register B

The Read Access Enable Register B controls read access for transactions to the protected memory region with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The DAM is prepared for an 6 bit TAG ID. The registers RGNACCENRAi / RGNACCENRBi are providing one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to RGNACCENRBi.ENn: EN0 -> TAG ID 100000<sub>B</sub>, EN1 -> TAG ID 100001<sub>B</sub>, ..., EN31 -> TAG ID 111111<sub>B</sub>. Note that the accesses from the Cerberus, HSM and IOC32 will always be permitted regardless of the value in the register.

### RGNACCENRBx (x=0-7)

**DAM Region Read Enable Register B** (000DC<sub>H</sub>+x\*10<sub>H</sub>) **Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN63</b>	<b>EN62</b>	<b>EN61</b>	<b>EN60</b>	<b>EN59</b>	<b>EN58</b>	<b>EN57</b>	<b>EN56</b>	<b>EN55</b>	<b>EN54</b>	<b>EN53</b>	<b>EN52</b>	<b>EN51</b>	<b>EN50</b>	<b>EN49</b>	<b>EN48</b>
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EN47</b>	<b>EN46</b>	<b>EN45</b>	<b>EN44</b>	<b>EN43</b>	<b>EN42</b>	<b>EN41</b>	<b>EN40</b>	<b>EN39</b>	<b>EN38</b>	<b>EN37</b>	<b>EN36</b>	<b>EN35</b>	<b>EN34</b>	<b>EN33</b>	<b>EN32</b>
rw															

Field	Bits	Type	Description
<b>ENn (n=32-63)</b>	n-32	rw	<p><b>Access Enable for Master TAG ID n</b></p> <p>This bit enables read access to the module kernel addresses for transactions with the Master TAG ID n</p> <p><math>0_B</math> No read accesses are permitted for this region. Read accesses will terminate with an error condition</p> <p><math>1_B</math> Read permitted for this region</p>

## Default Application Memory (DAM)

### 8.4 Revision History

**Table 243 Revision History**

Reference	Change to Previous Version	Comment
<b>V1.3.10</b>		
<a href="#">Page 12</a>	Revision history clean up, no functional changes.	-
<b>V1.3.11</b>		
<a href="#">Page 1</a>	<a href="#">Section 8.1</a> . Overlay memory removed from feature list. This is not supported by the overlay configuration options.	0000056731-35
<a href="#">Page 1</a>	<a href="#">Section 8.2.1</a> . Reference to overlay memory removed. This is not supported by the overlay configuration options.	0000056731-35

---

## System Control Units (SCU)

### 9 System Control Units (SCU)

The System Control Unit (SCU) is a cluster of sub-modules which control various system functions, including:

- Reset Control (RCU)
- Trap generation (TR)
- System Registers for miscellaneous functions (SRU)
- Watchdog Timers (WDT)
- External Request handling (ERU)
- Emergency Stop (ES)
- Power Management Control (PMC)

These submodules share a common bus interface.

## System Control Units (SCU)

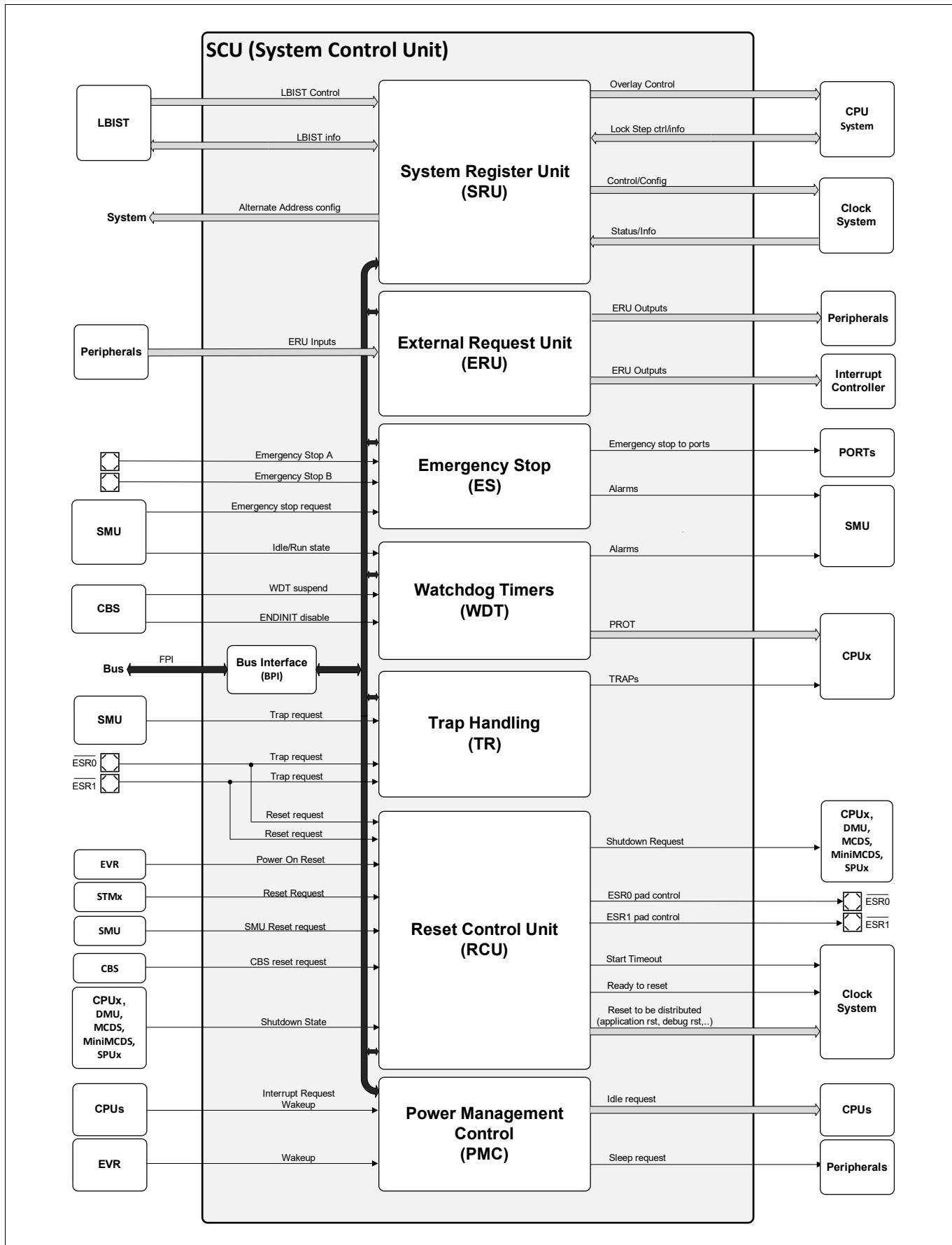


Figure 66 SCU Internal Structure Overview

## System Control Units (SCU)

### 9.1 Reset Control Unit (RCU)

This chapter contains the following sections:

- Basic Reset Operation (see [Section 9.1.2](#))
- External Reset sources and indications (see [Section 9.1.4](#))
- Boot Software Interface (see [Section 9.1.5](#))

#### 9.1.1 Feature List

- A complete device reset can be triggered by the primary voltage monitors
- Partial device resets can be triggered by assertion of the PORST, ESR0 or ESR1 pin(s)
- Partial device resets can be triggered in response to a Safety Alarm
- Partial device resets can be triggered by software
- Partial device resets can be triggered by test or JTAG
- Partial device resets can be triggered by any System Timer

Partial reset definition can be extracted from [Table 244](#) (reset sources to reset type) together with [Table 245](#) (reset type to which part of the device is affected).

This module has the same features and functionality in all TC3xx devices.

#### 9.1.1.1 Delta to AURIX

The TC3xx reset concept is based upon the reset concept of the TC2xx product range. The most significant changes are:

- Each CPU can be individually reset using CPUx\_KRSTy registers (See CPU chapter)
- Several registers formerly protected by SE are now only protected by E (to allow easier changes during Safety Applications)

#### 9.1.2 Overview

This section describes the conditions under which the AURIX™ TC3xx Platform will be reset and the reset operation configuration and control.

#### 9.1.2.1 Reset Triggers

The following reset request triggers are available:

- Supply monitor (SWD) triggers a power-on reset (cold reset)
- Core voltage EVR (EVRC) monitor triggers a power-on reset (cold reset)
- 3.3V EVR monitor triggers a power-on reset (cold reset) (If product has EVR33)
- Standby EVR (STBYR) monitor triggers a power-on reset (cold reset)
- External active low hardware “power-on” reset request trigger; PORST (can be either a warm reset or to extend a cold reset)
- External System Request reset trigger pins;  $\overline{\text{ESR0}}$  and  $\overline{\text{ESR1}}$  (warm reset)
- Safety Management Unit (SMU) alarm reset request trigger, (warm reset)
- Software reset (SW), (warm reset)
- System Timer (STMx) trigger (warm reset)
- Resets via the JTAG interface
- Resets initiated via On-Chip Debug System (OCDS)

## System Control Units (SCU)

- Software triggered module reset

*Note:* The JTAG resets are described in the OCDS chapter.

### 9.1.2.2 Reset Types

The following list describes the different reset types.

- Power-on Reset:

This reset results in initialization of the complete system into a defined state. A Power-on Reset also generates a Debug Reset and a System Reset and therefore also an Application Reset. (See also the section about Warm and Cold Resets)

- System Reset:

This reset leads to an initialization into a defined state of the complete system but without a reset of the power subsystem, debug subsystem or reset configuration registers.

A System Reset also generates an Application Reset.

- Debug Reset:

This reset leads to an initialization into a defined state of the complete debug system.

- Application Reset:

This reset leads to an initialization into a defined state of the complete application system with the following parts: all peripherals, the CPUs and parts of the SCU.

- Module Resets:

Module resets result in individual modules being initialized into a defined state without any impact on the rest of the system.

### 9.1.2.3 Reset Sources Overview

The connection of the reset sources and the activated reset signals/domains are shown in [Table 244](#).

**Table 244 Effect of Reset Triggers**

Reset Request Trigger	Application Reset	Debug Reset	System Reset
<b>PORST</b>	Activated	Activated	Activated
<b>STBYR</b>	Activated	Activated	Activated
<b>SWD</b>	Activated	Activated	Activated
<b>EVRC</b>	Activated	Activated	Activated
<b>EVR33</b>	Activated	Activated	Activated
<b>ESR0</b>	Configurable	Not Activated	Configurable
<b>ESR1</b>	Configurable	Not Activated	Configurable
<b>SMU</b>	Configurable	Not Activated	Configurable
<b>STMx</b>	Configurable	Not Activated	Configurable
<b>SW</b>	Configurable	Not Activated	Configurable
<b>Cerberus RSTCL0<sup>1)</sup></b>	Activated	Not Activated	Activated
<b>Cerberus RSTCL1<sup>2)</sup></b>	Not Activated	Activated	Not Activated
<b>Cerberus RSTCL3<sup>3)</sup></b>	Activated	Not Activated	Not Activated
<b>Module Resets<sup>4)</sup></b>	Not Activated	Not Activated	Not Activated

1) Cerberus resets may be triggered by CBS\_OCNTRL or CBS\_OJCONF

## System Control Units (SCU)

- 2) Cerberus resets may be triggered by CBS\_OCNTRL or CBS\_OJCONF
- 3) Cerberus resets may be triggered by CBS\_OCNTRL or CBS\_OJCONF
- 4) Module Resets (via MOD\_KRSTx.RST register bits in some modules) have no effect on chip-level reset system. The scope of a module reset is limited to the module itself.

### 9.1.2.4 Warm and Cold Resets

A “Warm reset” is a reset which is triggered while the system is already operational and the supplies remain stable. It is used to return the system to the same known state. On a warm reset request, any reset of pin states will take place immediately, but the internal circuitry will only be reset after the system has been brought into a state where memory contents and debug trace data will not be corrupted and the current consumption has been ramped down. Note that PORST may be asserted as a warm reset. In this specification a register denoted as “Power-on Reset” shall also be reset on a warm PORST without a power-on event.

A “Cold Power-on Reset” is a reset which is triggered for the first time during a system power-up or in response to a temporary power failure. During the power-up the EVR primary undervoltage monitors will trigger a complete reset of the system, placing the system into a known state. The PORST pin can be used to extend this reset phase and control the timing of its release. The pins and internal states are placed immediately into their reset state when the trigger is asserted.

Some special registers in the EVR are associated with functionality which should not be re-initialized even in the case of a temporary violation of an undervoltage monitor but only on a real power-up event. These are reset only when the supply drops below a very low threshold. The reset for these registers is described as “EVR Power-On Reset”

### 9.1.2.5 EVR Resets and PORST

The PORST pin is a bidirectional reset in/output intended for external triggering of power-related resets. If this pin is left open then the default behavior shall be that the device remains in a reset state. If the PORST pin remains asserted after a power event then the reset will be extended until it is de-asserted. This does not replace the ESR pins functional reset.

### 9.1.2.6 Module Reset Behavior

**Table 245** lists how the various functions of the AURIX™ TC3xx Platform are affected by each reset type. An “X” means that this block has at least some register/bits that are affected by the corresponding reset.

**Table 245 Effect of Reset on Device Functions**

Module / Function	Application Reset	Debug Reset	System Reset	Warm “Power-on” Reset	Cold Power-on Reset
<b>CPUx<sup>1)</sup></b>	X	X	X	X	X
<b>Peripherals (except SCU)</b>	X	X	X	X	X
<b>SCU</b>	X	Not affected	X	X	X
<b>Flash Memory</b>	Not affected	Not affected, reliable	X	X	X
<b>JTAG Interface</b>	Not affected	Not affected	Not affected	X	X
<b>OCDS</b>	Not affected	X	Not affected	X	X
<b>MCDS</b>	Not affected	X	Not affected	X	X
<b>Backup Clock</b>	Not affected	Not affected	Not affected	Not affected	Not affected

## System Control Units (SCU)

**Table 245 Effect of Reset on Device Functions (cont'd)**

Module / Function	Application Reset	Debug Reset	System Reset	Warm “Power-on” Reset	Cold Power-on Reset
<b>XTAL Oscillator, PLLs</b>	Not affected	Not affected	X	X	X
<b>Port Pins</b>	X	Not affected	X	X	X
<b>Pins ESRx</b>	Not affected	Not affected	X	X	X
<b>EVR</b>	Not affected	Not affected	Not affected	Not affected	X
<b>Standby Controller</b>	Not affected	Not affected	Not affected	X	X
<b>On-chip Static RAMs<sup>2)</sup></b>					
<b>CAN</b>	Initialized	Initialized	Initialized	Initialized	Uninitialized
<b>All DSPRs<sup>3)</sup></b>	Not affected, reliable	Not affected, reliable	Not affected, reliable	Not affected, reliable	Uninitialized
<b>All PSPRs<sup>4)</sup></b>	Not affected, reliable	Not affected, reliable	Not affected, reliable	Not affected, reliable	Uninitialized
<b>All PCACHEs and DCACHEs<sup>5)</sup></b>	Cache invalidated	Cache invalidated	Cache invalidated	Cache invalidated	Uninitialized
<b>LMU<sup>6)</sup></b>	Not affected, reliable	Not affected, reliable	Not affected, reliable	Not affected, reliable	Uninitialized
<b>HSM<sup>7)</sup></b>	Not affected, reliable. Accessible only from HSM	Not affected, reliable. Accessible only from HSM	Not accessible until re-initialized	Not accessible until re-initialized	Not accessible until initialized
<b>Other</b>	Not affected, reliable	Not affected, reliable	Not affected, reliable	Not affected, reliable	Uninitialized

- 1) Lock-stepped checker cores are initialized to the same reset state as the main core to avoid comparator fails during start-up
- 2) Reliable here means that also the redundancy is not affected by the reset.
- 3) DSPR is partially used as a scratchpad by the startup firmware. Previous data stored can be overwritten on start-up. Please see firmware specification, chapter “RAM overwrite during start-up” for more detailed information. Initialization on reset can be optionally configured by DMU\_HF\_PROCONRAM.RAMIN and DMU\_HF\_PROCONRAM.RAMINSEL.
- 4) Depending upon DMU\_HF\_PROCONRAM.RAMIN and DMU\_HF\_RAMINSEL setting, these memories may be automatically erased on cold or warm resets.
- 5) Tag memories are invalidated by hardware at reset. Depending upon DMU\_HF\_PROCONRAM.RAMIN setting, cache content may also be automatically erased at reset (See MTU chapter for details).
- 6) Depending upon DMU\_HF\_PROCONRAM.RAMIN and LMUINSEL setting, these memories may be automatically erased on cold or warm resets. Initialization is disabled for DLMUs with standby capability if the standby is enabled.
- 7) HSM memories are accessible to the HSM module itself after application reset provided that access was already enabled prior to the reset. In this case the contents are reliable. In all other cases an automatic initialization will be triggered.

### 9.1.3 Reset Controller Functional Description

A reset is generated if an enabled reset request trigger is asserted. Most reset triggers can be configured to initiate different types of reset. No action (disabled) is a valid configuration which is selected by setting the corresponding bit field in the Reset Configuration Register to 00<sub>B</sub>.

## System Control Units (SCU)

A Debug Reset can only be requested by dedicated reset request triggers and can not be selected via a Reset Configuration Register . For more information see also register RSTCON.

### 9.1.3.1 Reset Generation

The figure below shows the RCU controlling the reset generation for the complete device, with the exception of the JTAG reset domain.

*Note:* The JTAG reset domain is controlled by the TRST pin.

The RCU detects the different reset requests, schedules a shutdown of the system, and then generates the appropriate internal reset pulse(s).

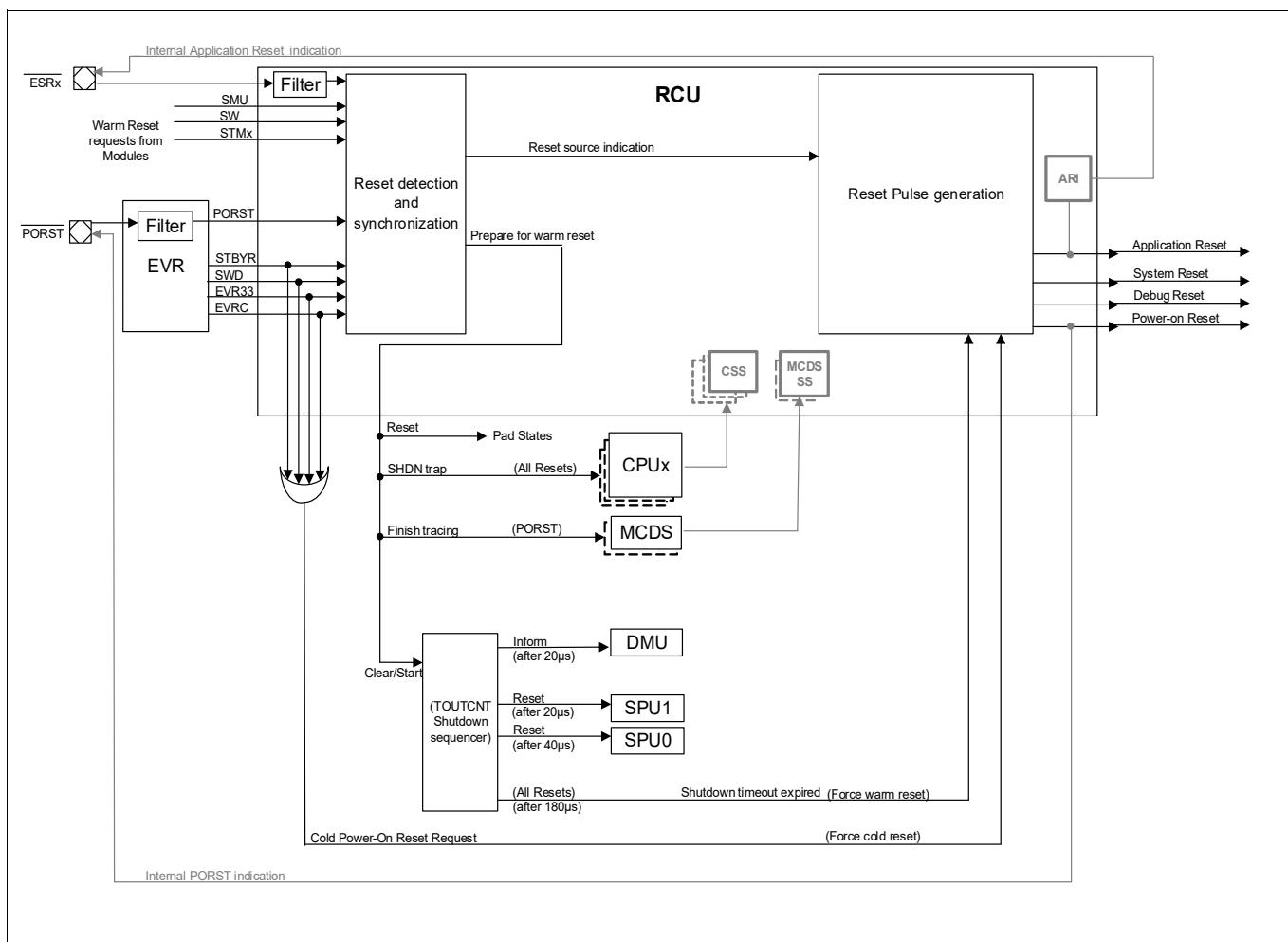


Figure 67 Reset Overview

### 9.1.3.2 Shutdown and Reset Delay Timeout Counter (TOUTCNT)

The system is automatically brought to a stable state before the internal reset(s) are applied so that the RAM content may be reliably reused after a warm reset and current jumps are controlled.

On detection of a warm Application Reset, System Reset or Power-On Reset trigger, the pads are put into their reset state and a Shutdown Trap request is sent by the RCU to the CPU(s). This request causes the CPU(s) which are not already in a halt or idle state to unconditionally execute an immediate ROM shutdown routine which executes a sequenced ramp-down of the device power consumption (to prevent current jumps which might trigger EVR reset). During this routine each CPU executes a WAIT instruction and stalls, to ensure that all ongoing

## System Control Units (SCU)

write transactions have been completed. Interrupts and NMI to the CPUs during the shutdown routine will not reawaken CPUs.

The internal reset starts when all relevant system modules are in the shutdown state. If timeout counter TOUTCNT exceeds a timeout period (180 microseconds) before the system become idle then the reset is started regardless. This timeout ensures that a reset would still occur in the case of an internal deadlock.

The RSTCON2.CSS bits indicate whether each CPU successfully flushed its write buffers and reached an idle state before the previous reset. These bits can therefore be used to determine whether RAM content integrity can be trusted after the previous reset cycle.

### 9.1.3.3 Reset Triggers

There are two types of reset triggers for the reset control logic:

- Triggers that lead to a specific reset
- Triggers that lead to a configurable reset

#### 9.1.3.3.1 Specific Reset Triggers

Assertion of these triggers leads to a predefined reset type. These triggers can not be enabled / disabled. All specific reset triggers are listed in [Table 246](#).

**Table 246 Specific Reset Triggers**

Reset Trigger	Reset Type Request
Cerberus 0 (CB0)	SystemReset
Cerberus 1 (CB1)	DebugReset
Cerberus 3 (CB3)	ApplicationReset
STBYR	Power-on Reset
SWD	Power-on Reset
EVRC	Power-on Reset
EVR33	Power-on Reset

#### 9.1.3.3.2 Configurable Reset Triggers

Assertion of these triggers leads to a configurable reset type. These triggers can be enabled / disabled. The result of the reset triggers is defined by the corresponding bit field in register RSTCON.

#### 9.1.3.3.3 Prevention of Double SMU Resets

The SMU can request a reset in response to an alarm. The type (Application or System Reset) is selected via register RSTCON.SMU. After boot firmware initialization, the default behaviour of the SMU/RCU is that a Watchdog timeout would result in an NMI followed by an Application Reset. If an SMU-induced Application Reset occurs twice, a severe system malfunction is assumed and the AURIX™ TC3xx Platform is held in permanent Application Reset until a Power-On or System Reset occurs. This prevents the device from being periodically reset if the application fails to service the watchdog correctly and the watchdog repeatedly times out.

An internal flag is set when the first SMU reset is requested. If a second reset is also requested by the SMU when the internal flag is already set, the double SMU reset event has occurred and a permanent reset request is automatically generated. This internal flag is only reset by a System Reset or when bit SMU\_WDTSCON1.CLRIRF is set and bit SMU\_WDTSCON0.ENDINIT has also been set. A correct service of the WDT does not automatically clear this flag.

## System Control Units (SCU)

If this behaviour is undesirable, then the CLRIRF bit should be written by the application during initialization (before the watchdog times out), to clear the flag and prevent any subsequent occurrence of a permanent reset.

**Note:** *If for any reason random code is executed bit field RSTCON.SMU can be updated unintentionally. This can result in an SMU alarm not leading to a reset. To avoid this after the SSW is finished this bit field should be checked and the Safety WDT ENDINIT protection enabled.*

### 9.1.3.4 Debug Reset Specific Behavior

For safety reasons it is required by the debugger that if the OCDS system is disabled a DebugReset is also asserted every time an Application Reset is asserted.

### 9.1.3.5 Module Resets

Many modules can be reset individually. The module reset has no effect outside the module itself. A module reset can only be triggered by writing ‘1’ into both of the module reset registers MOD\_KRST1.RST and register MOD\_KRST0.RST.

The Module Reset register bits may only be written by those masters which have been explicitly configured to have module access (See module ACCEN register). In addition, it is only possible to write to these reset bits during the short time window after a correct ENDINIT password unlock sequence (See Watchdog chapter for details). This prevents accidental module resets by rogue software.

The table below shows the modules which have Module Reset capability

**Table 247 Module Resets**

Module	Module Reset capability
CANx	Module reset implemented in CAN
ASCLINx	Module reset implemented in ASCLIN
GTM	Module reset implemented in GTM
ERAYx	Module reset implemented in ERAY
QSPIx	Module reset implemented in QSPI
HSSL	Module reset implemented in HSSL
ETHERMAC	Module reset implemented in ETHERMAC
MSCx	Module reset implemented in MSC
SENTx	Module reset implemented in SENT
VADC	Module reset implemented in VADC
EDSADC	Module reset implemented in EDSADC
SCR	Module reset implemented in PMS (PMSWCR1.SCRSTREQ)
CCU6	Module reset implemented in CCU6
GPT12	Module reset implemented in GPT12
STMx	Module reset implemented in STMx
PSI5	Module reset implemented in PSI5
PSI5-S	Module reset implemented in PSI5-S
DMA	Per channel reset implemented in DMA
SCU, RCU, PMS, CCU	No module reset capability
PMU, Flash	No module reset capability

## System Control Units (SCU)

**Table 247 Module Resets (cont'd)**

Module	Module Reset capability
LMU	No module reset capability
POTS	No module reset capability
IR	No module reset capability
IOM	Module reset implemented in IOM
HSM	No module reset capability
I2Cx	Module reset implemented in I2C
CPUx	Individual CPU reset capability
SPU	Module reset implemented in SPU
RIF	Module reset implemented in RIF
HSPDM	Module reset implemented in HSPDM
FCE	Module reset implemented in FCE
SDMMC	Module reset implemented in SDMMC

### 9.1.3.5.1 CPU Module Resets

A CPU Module reset has the same effect on an individual CPU as an Application Reset, but without a full reinitialization of other system elements outside the CPU

- CPU0 - Jumps to application start after a Module Reset
- Other CPUs - Wait in HALT mode after Module Reset

The CPUx\_KRST.STATUS registers can be used to determine whether the last CPUx reset was caused by a Module Reset or a chip-level reset. It is recommended that this register should be cleared by the application after reading. See CPU and Start-Up Chapters for more details.

### 9.1.3.6 Reset Controller Registers

#### 9.1.3.6.1 Status Registers

After a chip level reset has been executed, the Reset Status registers provide information on the trigger of the last reset(s). Warm reset status bits are updated upon each reset cycle. A reset cycle is finished when all resets are deasserted. Within a reset cycle the status flags are used as sticky flags. Module level resets have no effect on the Reset Status register. Bits which may indicate a cold reset trigger (STBYR, SWD,EVR33, PORST and EVRC and ) provide useful information to the application and are not cleared automatically. The application may clear these bits by writing to bit RSTCON2.CLRC.

*Note:* All bits of this register are implemented in the IP. The product specific names are purely defined by the product integration.

**System Control Units (SCU)****Reset Status Register****RSTSTAT****Reset Status Register**(0050<sub>H</sub>)Reset Value: [Table 248](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	<b>LBTER M</b>	<b>LBPO RST</b>	<b>STBYR</b>	<b>HSMA</b>	<b>HSMS</b>	<b>SWD</b>	<b>EVR33</b>	<b>EVRC</b>	<b>R22</b>	<b>R21</b>	<b>CB3</b>	<b>CB1</b>	<b>CB0</b>	0	<b>PORS T</b>
r	rh	rh	rh	rh	rh	rh	rh	rh	rX	rX	rh	rh	rh	r	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				0	<b>STM5</b>	<b>STM4</b>	<b>STM3</b>	<b>STM2</b>	<b>STM1</b>	<b>STM0</b>	<b>SW</b>	<b>SMU</b>	0	<b>ESR1</b>	<b>ESR0</b>
				r	rh	rh	rh	rh	rh	rh	rh	rh	r	rh	rh

Field	Bits	Type	Description
<b>ESR0</b>	0	rh	<b>Reset Request Trigger Reset Status for ESR0</b> 0 <sub>B</sub> The last reset was not requested by this reset trigger 1 <sub>B</sub> The last reset was requested by this reset trigger
<b>ESR1</b>	1	rh	<b>Reset Request Trigger Reset Status for ESR1</b> 0 <sub>B</sub> The last reset was not requested by this reset trigger 1 <sub>B</sub> The last reset was requested by this reset trigger
<b>SMU</b>	3	rh	<b>Reset Request Trigger Reset Status for SMU</b> (See SMU section for SMU trigger sources, including Watchdog Timers) 0 <sub>B</sub> The last reset was not requested by this reset trigger 1 <sub>B</sub> The last reset was requested by this reset trigger
<b>SW</b>	4	rh	<b>Reset Request Trigger Reset Status for SW</b> 0 <sub>B</sub> The last reset was not requested by this reset trigger 1 <sub>B</sub> The last reset was requested by this reset trigger
<b>STM0</b>	5	rh	<b>Reset Request Trigger Reset Status for STM0 Compare Match</b> 0 <sub>B</sub> The last reset was not requested by this reset trigger 1 <sub>B</sub> The last reset was requested by this reset trigger
<b>STM1</b>	6	rh	<b>Reset Request Trigger Reset Status for STM1 Compare Match (If Product has STM1)</b> 0 <sub>B</sub> The last reset was not requested by this reset trigger 1 <sub>B</sub> The last reset was requested by this reset trigger
<b>STM2</b>	7	rh	<b>Reset Request Trigger Reset Status for STM2 Compare Match (If Product has STM2)</b> 0 <sub>B</sub> The last reset was not requested by this reset trigger 1 <sub>B</sub> The last reset was requested by this reset trigger
<b>STM3</b>	8	rh	<b>Reset Request Trigger Reset Status for STM3 Compare Match (If Product has STM3)</b> 0 <sub>B</sub> The last reset was not requested by this reset trigger 1 <sub>B</sub> The last reset was requested by this reset trigger

**System Control Units (SCU)**

Field	Bits	Type	Description
<b>STM4</b>	9	rh	<b>Reset Request Trigger Reset Status for STM4 Compare Match (If Product has STM4)</b> 0 <sub>B</sub> The last reset was not requested by this reset trigger 1 <sub>B</sub> The last reset was requested by this reset trigger
<b>STM5</b>	10	rh	<b>Reset Request Trigger Reset Status for STM5 Compare Match (If Product has STM5)</b> 0 <sub>B</sub> The last reset was not requested by this reset trigger 1 <sub>B</sub> The last reset was requested by this reset trigger
<b>PORST</b>	16	rh	<b>Reset Request Trigger Reset Status for PORST</b> This bit is also set if the bits CB0, CB1, and CB3 are set in parallel. 0 <sub>B</sub> This reset trigger has not occurred since the last clear (by RSTCON2.CLRC) 1 <sub>B</sub> This reset trigger has occurred since the last clear (by RSTCON2.CLRC)
<b>CB0</b>	18	rh	<b>Reset Request Trigger Reset Status for Cerberus System Reset</b> 0 <sub>B</sub> The last reset was not requested by this reset trigger 1 <sub>B</sub> The last reset was requested by this reset trigger
<b>CB1</b>	19	rh	<b>Reset Request Trigger Reset Status for Cerberus Debug Reset</b> 0 <sub>B</sub> The last reset was not requested by this reset trigger 1 <sub>B</sub> The last reset was requested by this reset trigger
<b>CB3</b>	20	rh	<b>Reset Request Trigger Reset Status for Cerberus Application Reset</b> 0 <sub>B</sub> The last reset was not requested by this reset trigger 1 <sub>B</sub> The last reset was requested by this reset trigger
<b>R21</b>	21	rX	<b>Reserved - 0</b> Read as 0; should be written with 0.
<b>R22</b>	22	rX	<b>Reserved - 0</b> Read as 0; should be written with 0.
<b>EVRC</b>	23	rh	<b>Reset Request Trigger Reset Status for EVRC</b> 0 <sub>B</sub> This reset trigger has not occurred since the last clear (by RSTCON2.CLRC) 1 <sub>B</sub> This reset trigger has occurred since the last clear (by RSTCON2.CLRC)
<b>EVR33</b>	24	rh	<b>Reset Request Trigger Reset Status for EVR33</b> 0 <sub>B</sub> This reset trigger has not occurred since the last clear (by RSTCON2.CLRC) 1 <sub>B</sub> This reset trigger has occurred since the last clear (by RSTCON2.CLRC)
<b>SWD</b>	25	rh	<b>Reset Request Trigger Reset Status for Supply Watchdog (SWD)</b> The Supply Watchdog trigger is described in Power Management Controller “Supply Monitoring” chapter 0 <sub>B</sub> This reset trigger has not occurred since the last clear (by RSTCON2.CLRC) 1 <sub>B</sub> This reset trigger has occurred since the last clear (by RSTCON2.CLRC)

**System Control Units (SCU)**

Field	Bits	Type	Description
<b>HSMS</b>	26	rh	<b>Reset Request Trigger Reset Status for HSM System Reset (HSM S)</b> $0_B$ The last reset was not requested by this reset trigger $1_B$ The last reset was requested by this reset trigger
<b>HSMA</b>	27	rh	<b>Reset Request Trigger Reset Status for HSM Application Reset (HSM A)</b> $0_B$ The last reset was not requested by this reset trigger $1_B$ The last reset was requested by this reset trigger
<b>STBYR</b>	28	rh	<b>Reset Request Trigger Reset Status for Standby Regulator Watchdog (STBYR)</b> $0_B$ This reset trigger has not occurred since the last clear (by RSTCON2.CLRC) $1_B$ This reset trigger has occurred since the last clear (by RSTCON2.CLRC)
<b>LBPORST</b>	29	rh	<b>LBIST termination due to PORST</b> This bitfield indicates if the LBIST was early terminated due to the occurrence of a Power On Reset. If the status of this bitfield is 0, the application must still check the LBTERM to check if the LBIST was terminated properly. This bitfield is cleared when the RSTCON2.CLRC is set. $0_B$ LBIST was not terminated early due to a Power On Reset $1_B$ LBIST early termination due to the occurrence of Power On Reset
<b>LBTERM</b>	30	rh	<b>LBIST was properly terminated</b> This bitfield indicates if the LBIST was terminated properly. This bitfield is cleared when the RSTCON2.CLRC is set. $0_B$ LBIST was not terminated properly $1_B$ LBIST was terminated properly
<b>0</b>	2, 15:11, 17, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 248 Reset Values of RSTSTAT**

Reset Type	Reset Value	Note
Cold PowerOn Reset	$0XX1\ 0000_H$	RSTSTAT
Cold PowerOn Reset	$1001\ 0000_H$	RSTSTAT (Triggered by LVD Reset)

## System Control Units (SCU)

## 9.1.3.6.2 Reset Configuration Registers

## Reset Configuration Register

RSTCON

## Reset Configuration Register

(0058<sub>H</sub>)Reset Value: [Table 249](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							0				STM5		STM4		STM3
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	STM2		STM1		STM0		SW		SMU		0		ESR1		ESR0
	rw		rw		rw		rw		rw		rw		rw		rw

Field	Bits	Type	Description
ESR0	1:0	rw	<b>ESR0 Reset Request Trigger Reset Configuration</b> This bit field defines which reset is generated by a reset request trigger from ESR0 reset. 00 <sub>B</sub> No reset is generated for a trigger of ESR0 01 <sub>B</sub> A System Reset is generated for a trigger of ESR0 reset 10 <sub>B</sub> An Application Reset is generated for a trigger of ESR0 reset 11 <sub>B</sub> Reserved, do not use this combination
ESR1	3:2	rw	<b>ESR1 Reset Request Trigger Reset Configuration</b> This bit field defines which reset is generated by a reset request trigger from ESR1 reset. 00 <sub>B</sub> No reset is generated for a trigger of ESR1 01 <sub>B</sub> A System Reset is generated for a trigger of ESR1 reset 10 <sub>B</sub> An Application Reset is generated for a trigger of ESR1 reset 11 <sub>B</sub> Reserved, do not use this combination
SMU	7:6	rw	<b>SMU Reset Request Trigger Reset Configuration</b> This bit field defines which reset is generated by a reset request trigger from SMU reset. 00 <sub>B</sub> No reset is generated for a trigger of SMU 01 <sub>B</sub> A System Reset is generated for a trigger of SMU reset 10 <sub>B</sub> An Application Reset is generated for a trigger of SMU reset 11 <sub>B</sub> Reserved, do not use this combination
SW	9:8	rw	<b>SW Reset Request Trigger Reset Configuration</b> This bit field defines which reset is generated by a reset request trigger from software reset. 00 <sub>B</sub> No reset is generated for a trigger of software reset 01 <sub>B</sub> A System Reset is generated for a trigger of Software reset 10 <sub>B</sub> An Application Reset is generated for a trigger of Software reset 11 <sub>B</sub> Reserved, do not use this combination

**System Control Units (SCU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>STM0</b>	11:10	rw	<p><b>STM0 Reset Request Trigger Reset Configuration</b></p> <p>This bit field defines which reset is generated by a reset request trigger from STM0 compare match reset.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No reset is generated for an STM0 trigger</li> <li>01<sub>B</sub> A System Reset is generated for a trigger of STM0 reset</li> <li>10<sub>B</sub> An Application Reset is generated for a trigger of STM0 reset</li> <li>11<sub>B</sub> Reserved, do not use this combination</li> </ul>
<b>STM1</b>	13:12	rw	<p><b>STM1 Reset Request Trigger Reset Configuration (If Product has STM1)</b></p> <p>This bit field defines which reset is generated by a reset request trigger from STM1 compare match reset.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No reset is generated for a trigger of STM1</li> <li>01<sub>B</sub> A System Reset is generated for a trigger of STM1 reset</li> <li>10<sub>B</sub> An Application Reset is generated for a trigger of STM1 reset</li> <li>11<sub>B</sub> Reserved, do not use this combination</li> </ul>
<b>STM2</b>	15:14	rw	<p><b>STM2 Reset Request Trigger Reset Configuration (If Product has STM2)</b></p> <p>This bit field defines which reset is generated by a reset request trigger from STM2 compare match reset.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No reset is generated for a trigger of STM2</li> <li>01<sub>B</sub> A System Reset is generated for a trigger of STM2 reset</li> <li>10<sub>B</sub> An Application Reset is generated for a trigger of STM2 reset</li> <li>11<sub>B</sub> Reserved, do not use this combination</li> </ul>
<b>STM3</b>	17:16	rw	<p><b>STM3 Reset Request Trigger Reset Configuration (If Product has STM3)</b></p> <p>This bit field defines which reset is generated by a reset request trigger from STM3 compare match reset.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No reset is generated for an STM3 trigger</li> <li>01<sub>B</sub> A System Reset is generated for a trigger of STM3 reset</li> <li>10<sub>B</sub> An Application Reset is generated for a trigger of STM3 reset</li> <li>11<sub>B</sub> Reserved, do not use this combination</li> </ul>
<b>STM4</b>	19:18	rw	<p><b>STM4 Reset Request Trigger Reset Configuration (If Product has STM4)</b></p> <p>This bit field defines which reset is generated by a reset request trigger from STM4 compare match reset.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No reset is generated for a trigger of STM4</li> <li>01<sub>B</sub> A System Reset is generated for a trigger of STM4 reset</li> <li>10<sub>B</sub> An Application Reset is generated for a trigger of STM4 reset</li> <li>11<sub>B</sub> Reserved, do not use this combination</li> </ul>
<b>STM5</b>	21:20	rw	<p><b>STM5 Reset Request Trigger Reset Configuration (If Product has STM5)</b></p> <p>This bit field defines which reset is generated by a reset request trigger from STM5 compare match reset.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No reset is generated for a trigger of STM5</li> <li>01<sub>B</sub> A System Reset is generated for a trigger of STM5 reset</li> <li>10<sub>B</sub> An Application Reset is generated for a trigger of STM5 reset</li> <li>11<sub>B</sub> Reserved, do not use this combination</li> </ul>

## System Control Units (SCU)

Field	Bits	Type	Description
0	5:4, 31:22	rw	<b>Reserved</b> Should be written with 0.

**Table 249 Reset Values of RSTCON**

Reset Type	Reset Value	Note
PowerOn Reset	0000 0282 <sub>H</sub>	RSTCON

### Application Reset Disable Register

#### ARSTDIS

<b>Application Reset Disable Register (005C<sub>H</sub>)</b>															
<b>PowerOn Reset Value: 0000 0000<sub>H</sub></b>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								0	rw						
r								r							

Field	Bits	Type	Description
<b>STM0DIS</b>	0	rw	<b>STM0 Disable Reset</b> This bit field defines if an Application Reset leads to an reset for the STM0.  0 <sub>B</sub> An Application Reset resets the STM0 1 <sub>B</sub> An Application Reset has no effect for the STM0
<b>STM1DIS</b>	1	rw	<b>STM1 Disable Reset</b> This bit field defines if an Application Reset leads to an reset for the STM1.  0 <sub>B</sub> An Application Reset resets the STM1 1 <sub>B</sub> An Application Reset has no effect for the STM1
<b>STM2DIS</b>	2	rw	<b>STM2 Disable Reset</b> This bit field defines if an Application Reset leads to an reset for the STM2.  0 <sub>B</sub> An Application Reset resets the STM2 1 <sub>B</sub> An Application Reset has no effect for the STM2
<b>STM3DIS</b>	3	rw	<b>STM3 Disable Reset</b> This bit field defines if an Application Reset leads to an reset for the STM3.  0 <sub>B</sub> An Application Reset resets the STM3 1 <sub>B</sub> An Application Reset has no effect for the STM3

## System Control Units (SCU)

Field	Bits	Type	Description
<b>STM4DIS</b>	4	rw	<b>STM4 Disable Reset</b> This bit field defines if an Application Reset leads to an reset for the STM4. 0 <sub>B</sub> An Application Reset resets the STM4 1 <sub>B</sub> An Application Reset has no effect for the STM4
<b>STM5DIS</b>	5	rw	<b>STM5 Disable Reset</b> This bit field defines if an Application Reset leads to an reset for the STM5. 0 <sub>B</sub> An Application Reset resets the STM5 1 <sub>B</sub> An Application Reset has no effect for the STM5
<b>0</b>	7:6	rw	<b>Reserved</b> Should be written with 0.
<b>0</b>	31:8	r	<b>Reserved</b> Read as 0; should be written with 0.

## Software Reset Configuration Register

This register controls the SW Reset operation.

As for other kernel registers, write access to the SWRSTCON register is configurable via SCU\_ACCENO and additionally is temporally restricted by ENDINIT. These restrictions can be used to provide protection against accidental reset by unauthorised CPUs or system bus masters.

### SWRSTCON

Software Reset Configuration Register (0060<sub>H</sub>) Reset Value: [Table 250](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
															r
RES															
															w r

Field	Bits	Type	Description
<b>SWRSTREQ</b>	1	w	<b>Software Reset Request</b> This bit is automatically cleared and read always as zero. 0 <sub>B</sub> No SW Reset is requested 1 <sub>B</sub> A SW Reset request trigger is generated
<b>RES</b>	15:8	rw	<b>Reserved</b> Should be written with original content.
<b>0</b>	0, 7:2, 31:16	r	<b>Reserved</b> Read as 0; should be written with 0.

## System Control Units (SCU)

**Table 250 Reset Values of SWRSTCON**

Reset Type	Reset Value	Note
PowerOn Reset	0000 0000 <sub>H</sub>	SWRSTCON

### Additional Reset Control Register

#### RSTCON2

**Additional Reset Control Register (0064<sub>H</sub>) Reset Value: Table 251**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
USRINFO															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>0</b>	<b>0</b>	<b>0</b>	<b>CSSx</b>						<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>CLRC</b>	<b>FRT0</b>
r	r	r	rh						r	r	r	r	r	w	rw

Field	Bits	Type	Description
<b>FRT0</b>	0	rw	<b>Force Reset Timeout</b> 0 <sub>B</sub> Start next warm reset internally as soon as Flash and CPUs are idle and ready to reset 1 <sub>B</sub> Start next warm reset internally only when TOUTCNT expires
<b>CLRC</b>	1	w	<b>Clear Cold Reset Status</b> This bit simultaneously clears the sticky status bits which may indicate any previous cold reset (i.e. RSTSTAT.STBYR, RSTSTAT.SWD, RSTSTAT.EVR33, RSTSTAT.EVRC, RSTSTAT.PORST, RSTSTAT.LBPORST and RSTSTAT.LBTERM). 0 <sub>B</sub> No effect 1 <sub>B</sub> Clear cold reset RSTSTAT status bits
<b>CSSx</b>	12:7	rh	<b>CPU x Shutdown State Reached</b> The state of CPU x before the last warm reset. If any bit is zero after an Application Reset (or higher) then it is possible that SRAM content could have been corrupted by the reset. For products with fewer CPUs, only the LSBs are active and unused upper bits will always read '1'. For products with no MCDS/miniMCDS/MCDSlight, the bit will always read '1'. 00 <sub>H</sub> CPU x shutdown state not achieved prior to last reset 01 <sub>H</sub> CPU x in shutdown state at last reset
<b>USRINFO</b>	31:16	rw	<b>User Information</b> User data register (Cleared only on Cold Power-on reset). This may be used by an application to store information which must survive all warm resets

## System Control Units (SCU)

Field	Bits	Type	Description
0	2, 3, 4, 5, 6, 13, 14, 15	r	<b>Reserved</b> Internal use only. Bits may read as 0 or 1. Writes have no effect.

**Table 251 Reset Values of RSTCON2**

Reset Type	Reset Value	Note
Cold PowerOn Reset	0000 0000 <sub>H</sub>	RSTCON2

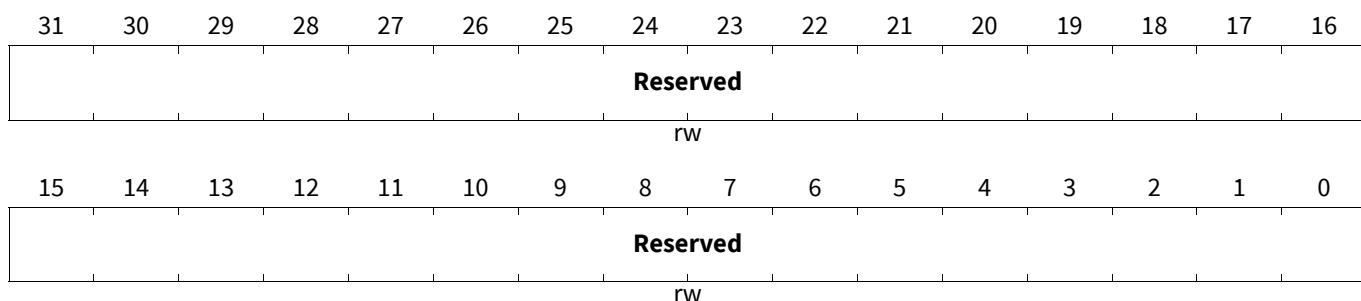
### Reset Configuration Register 3

**RSTCON3**

**Reset Configuration Register 3**

(0068<sub>H</sub>)

**Reset Value: Table 252**



Field	Bits	Type	Description
<b>Reserved</b>	31:0	rw	<b>Reserved</b> Write only with read value.

**Table 252 Reset Values of RSTCON3**

Reset Type	Reset Value	Note
Cold PowerOn Reset	0000 0000 <sub>H</sub>	RSTCON3
After SSW execution	8FFF 3400 <sub>H</sub>	

## System Control Units (SCU)

### 9.1.4 External Reset Sources and Indications

ESR interface pins are provided to give an external indication of internal resets, and to provide a mechanism for external sources to trigger internal resets:

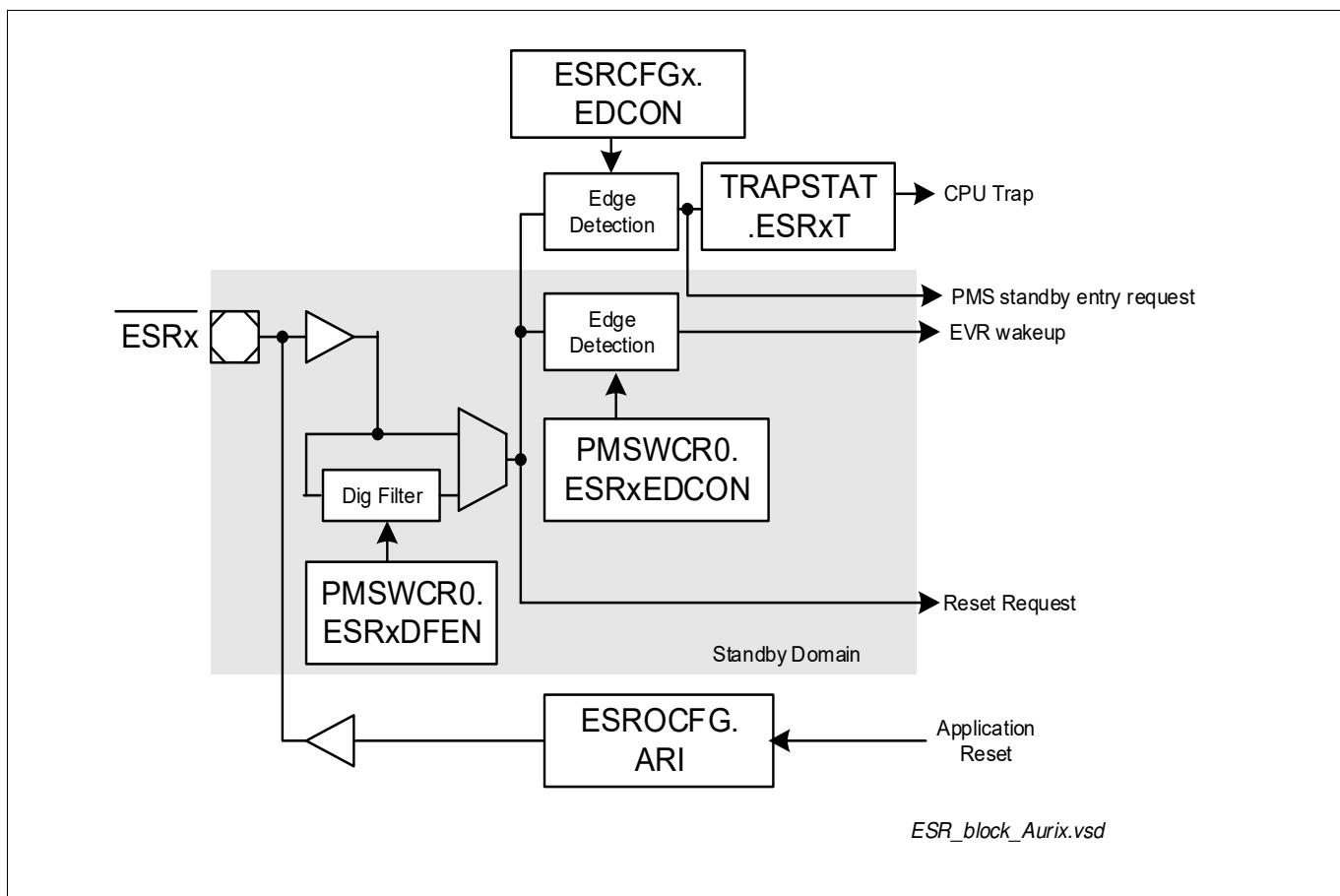
#### ESR pin functions

- Reset request trigger
- Wakeup trigger
- Reset indication output
- Trap request trigger

#### 9.1.4.1 External Service Requests (ESRx)

The ESR pins can be used in various ways:

- ESR inputs can trigger a reset
- ESR outputs can provide a reset indication
- ESR inputs can trigger a trap request
- ESR pins can be used as general data I/O pins
- ESR inputs can trigger a wake-up from standby mode



**Figure 68 ESR Operation**

## System Control Units (SCU)

### 9.1.4.1.1 ESRx as Reset Request Trigger

An ESR0/ESR1 pin reset request trigger can lead to a System or Application Reset. The type of the reset is configured via RSTCON.ESRx.

The input signals ESR0/ESR1 can be filtered. The filter can be disabled by register bit PMSWCR0.ESRxDFEN.

If the digital filter is enabled then pulses less than 30 ns cannot trigger a reset and pulses longer than 100ns will always result in a trigger..

The port pin behavior of ESR0/ESR1 pins can be configured by programming registers ESRCFG0/1.

The pad control functionality can be configured independently for each pin. The configuration comprises:

- Selection of driver type (open-drain or push-pull)
- Enable of the output driver (input and/or output capability)
- Enable of internal pull-up or pull-down resistance

By default at reset ESRx pads have pull-ups (but during LBIST the ESRx pin is changed to a weak pull-down)

### 9.1.4.1.2 ESRx as Reset Output

The external pins ESR0/ESR1 can provide a reset output indication (open drain) for Application Resets .

Register ESROCFG.ARI determines the output of the ESR pin(s) when one or more are configured as functional reset outputs. ARI is set automatically when any Application Reset starts and cleared by Boot or Application software (Note that an Application Reset also occurs on a System Reset or Power-On Reset). While the ARI bit is set, the configured ESR pin(s) drive active low. A subsequent write to ESROCFG.ARC clears the ARI bit, which deasserts the ESR output(s). The exact duration of the ESR pulse is determined by the value of the Flash Config Sector setting “ESR0CNT” which is copied by Boot Code into DMU\_HF\_PROCONDF.ESR0CNT on a cold power-on reset. The effect of this value is shown in **Table 253**. When the same ESR pin is configured as a PORT output, its state can instead be controlled directly by application software so that the application has the possibility to reset external devices.

**Table 253 ESRx Reset Indication Options**

ESR0CNT value	ESR Reset Indication Behaviour
0000 0000 <sub>H</sub>	ESR0 de-asserted as soon as possible after start of Boot Code execution
> 0000 0000 <sub>H</sub> and < FFF <sub>H</sub>	ESR0 de-asserted after programmed delay (by Boot Code). Boot Code may be extended and Application start may be delayed if programmed delay exceeds the normal Boot Code execution time. Programmed delay is ESR0CNT * 10 microseconds
FFF <sub>H</sub>	ESR0 not de-asserted by Boot Code. Application code must de-assert ESROCFG.ARI bit or PORT output to de-assert ESRx pin(s).

An external device may extend an existing reset condition by holding the ESR pin active.

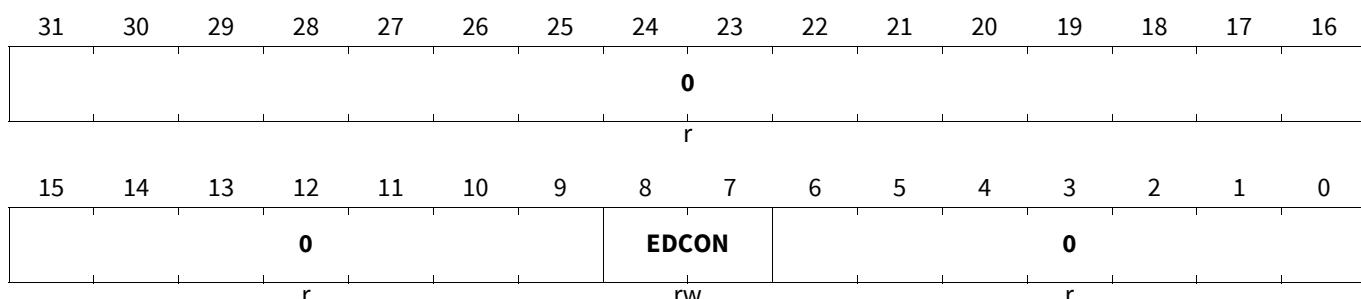
## System Control Units (SCU)

### 9.1.4.1.3 ESR Registers

#### ESRx Input Configuration Register

**ESRCFGx (x=0-1)**

**ESRx Input Configuration Register (0070<sub>H</sub>+x\*4)** **System Reset Value: 0000 0100<sub>H</sub>**



Field	Bits	Type	Description
<b>EDCON</b>	8:7	rw	<b>Edge Detection Control</b> This bit field defines the edges that lead to an ESRx trigger of the synchronous path. 00 <sub>B</sub> No trigger is generated 01 <sub>B</sub> A trigger is generated upon a rising edge 10 <sub>B</sub> A trigger is generated upon a falling edge 11 <sub>B</sub> A trigger is generated upon a rising OR falling edge
<b>0</b>	6:0, 31:9	r	<b>Reserved</b> Read as 0; should be written with 0.

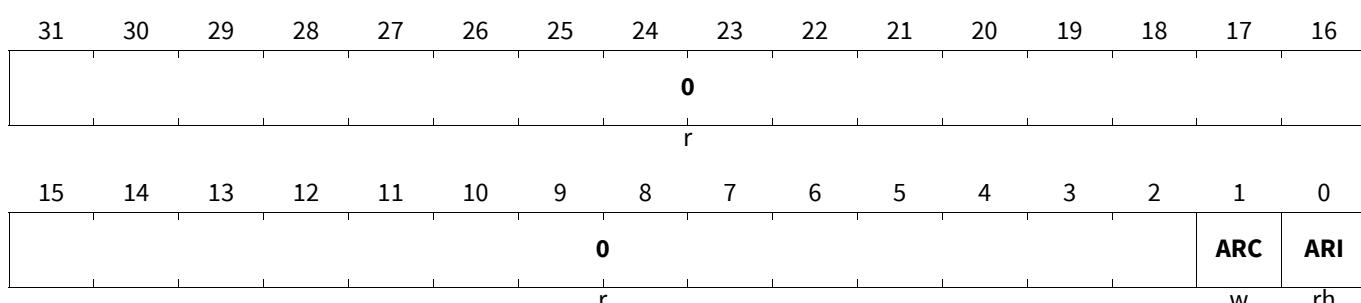
#### ESR Output Configuration Register

**ESROCFG**

**ESR Output Configuration Register**

(0078<sub>H</sub>)

**System Reset Value: 0000 000X<sub>H</sub>**



**System Control Units (SCU)**

Field	Bits	Type	Description
<b>ARI</b>	0	rh	<p><b>Application Reset Indicator</b></p> <p>This bit is set when an Application Reset request trigger occurs and cleared by writing to ARC.</p> <p>When the ARI bit is set and an ESR pin is configured as a reset output, the corresponding ESR input will not re-trigger a reset. This prevents feedback of the reset indication causing a new reset request.</p> <p>Extension of the reset by an external ESR source is handled by SSW.</p> <p><i>Note:</i> Observed reset value after boot will depend upon ARI mode.</p> <p><math>0_B</math> No application reset trigger detected (since last clear)  <math>1_B</math> Application reset trigger detected (since last clear)</p>
<b>ARC</b>	1	w	<p><b>Application Reset Indicator Clear</b></p> <p>Read as 0</p> <p><math>0_B</math> No effect  <math>1_B</math> Clear Application Reset Indicator (ARI)</p>
<b>0</b>	31:2	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

## System Control Units (SCU)

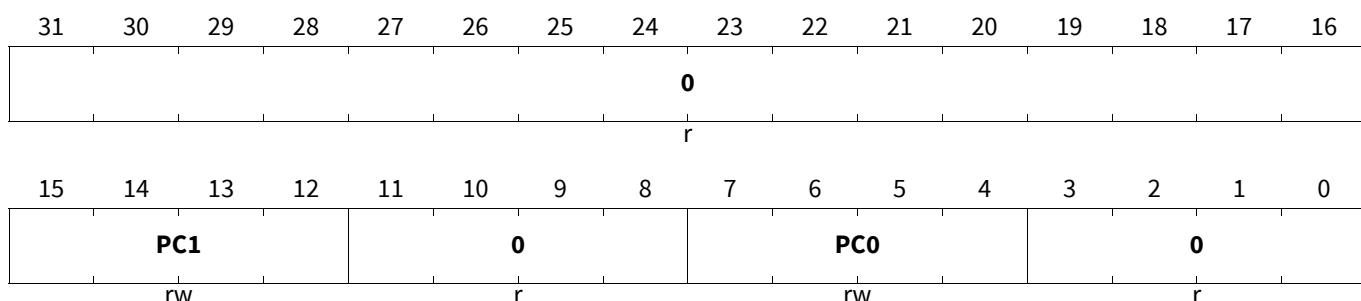
### Input/Output Control Register

The input/output control registers select the digital output and input driver functionality and characteristics of the pin. Direction (input or output), pull-up or pull-down devices for inputs, and push-pull or open-drain functionality for outputs can be selected by the corresponding bit fields PCx (x = 0-1).

### Input/Output Control Register

#### IOCR

**Input/Output Control Register (00A0<sub>H</sub>) System Reset Value: 0000 XOE0<sub>H</sub>**



Field	Bits	Type	Description
<b>PC0</b>	7:4	rw	<b>Control for ESR0 Pin</b> This bit field defines the ESR0 functionality according to the coding tables.
<b>PC1</b>	15:12	rw	<b>Control for ESR1 Pin</b> This bit field defines the ESR1 functionality according to the coding tables. The reset value of SCU_IOCR.PC1 is influenced by HWCFG6 and PMSWCR5.TRISTREQ. When a cold reset is activated and HWCFG6=1 then PC1 is reset to 2H and ESR1 will have input pull-up mode. If HWCFG6=0 then PC1 is reset to 0H and ESR1 will have tri-state mode. PC1 and the ESR1 reset state can also be configured by software with the PMSWCR5.TRISTREQ bit. PMSWCR5.TRISTREQ is not affected by warm reset or wake-up from standby so the IOCR.PC1 reset value is configured as per the state of the TRISTREQ bit prior to the warm reset
<b>0</b>	3:0, 11:8, 31:16	r	<b>Reserved</b> Read as 0; should be written with 0.

## System Control Units (SCU)

### Pad Control Coding

**Table 254** describes the coding of the PC0 bit field that determine the port line functionality.

**Table 254 PC0 Coding**

PC0[3:0]	I/O	Output Characteristics	Selected Pull-up/Pull-down/ Selected Output Function
$0X00_B$	Input is active and not inverted; Output is inactive		No input pull device connected
$0X01_B$			Input pull-down device connected
$0X10_B$			Input pull-up device connected
$0X11_B$			No input pull device connected
$1000_B$	Input is active and not inverted; Output is active	Push-pull	General-purpose Output
$1001_B$			Output drives a 0 for System Resets until ESROCFG.ARI is cleared, a weak pull-up is active otherwise
$1010_B$			Output drives a 0 for Application Resets until ESROCFG.ARI is cleared, a weak pull-up is active otherwise
$1011_B$			Reserved, do not use this combination
$1100_B$	the input is active and not inverted; Output is active	Open-drain	General-purpose Output
$1101_B$			Output drives a 0 for System Resets until ESROCFG.ARI is cleared, a weak pull-up is active otherwise
$1110_B$			Output drives a 0 for Application Resets until ESROCFG.ARI is cleared, a weak pull-up is active otherwise
$1111_B$			Reserved, do not use this combination

## System Control Units (SCU)

### Pad Control Coding

**Table 255** describes the coding of the PC1 bit field that determine the port line functionality.

**Table 255 PC1 Coding**

PC1[3:0]	I/O	Output Characteristics	Selected Pull-up/Pull-down/ Selected Output Function
0X00 <sub>B</sub>	Input is active and not inverted; Output is inactive		No input pull device connected
0X01 <sub>B</sub>			Input pull-down device connected
0X10 <sub>B</sub>			Input pull-up device connected
0X11 <sub>B</sub>			No input pull device connected
1000 <sub>B</sub>	Input is active and not inverted; Output is active	Push-pull	General-purpose Output
1001 <sub>B</sub>			Reserved, do not use this combination
1010 <sub>B</sub>			Output drives a 0 for Application Resets until ESROCFG.ARI is cleared, a 'Z' otherwise
1011 <sub>B</sub>			Reserved, do not use this combination
1100 <sub>B</sub>	the input is active and not inverted; Output is active	Open-drain	General-purpose Output
1101 <sub>B</sub>			Reserved, do not use this combination
1110 <sub>B</sub>			Output drives a 0 for Application Resets until ESROCFG.ARI is cleared, a 'Z' otherwise
1111 <sub>B</sub>			Reserved, do not use this combination

## System Control Units (SCU)

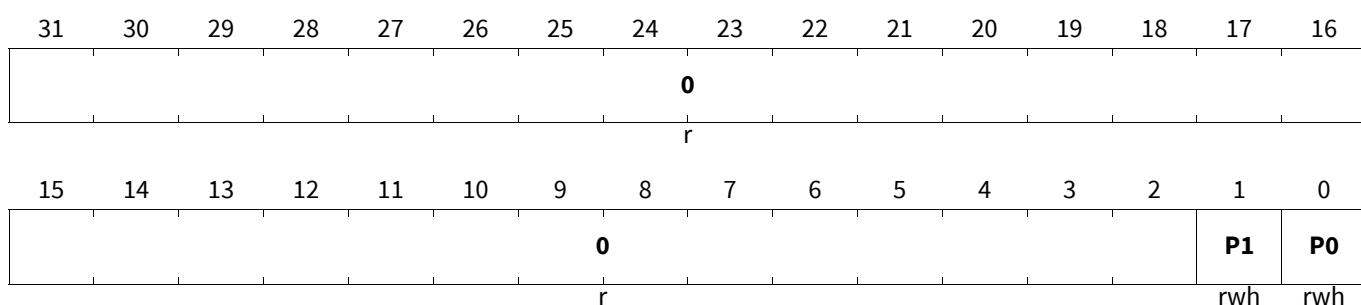
### ESR Output Register

The output register determines the value of a GPIO pin when it is selected by IOCR as output. Writing a 0 to a OUT.Px (x = 0-1) bit position delivers a low level at the corresponding output pin. A high level is output when the corresponding bit is written with a 1. Note that each single bit or group of bits of OUT.Px can be set/cleared by writing appropriate values into the output modification register OMR.

### ESR Output Register

#### OUT

#### ESR Output Register

(00A4<sub>H</sub>)System Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
Px (x=0-1)	x	rwh	<b>Output Bit x</b> This bit determines the level at the output pin <u>ESRx</u> if the output is selected as GPIO output. Px can also be set/cleared by control bits of the OMR register. 0 <sub>B</sub> The output level of <u>ESRx</u> is 0 1 <sub>B</sub> The output level of <u>ESRx</u> is 1
0	31:2	r	<b>Reserved</b> Read as 0; should be written with 0.

## System Control Units (SCU)

### ESR Output Modification Register

The output modification register contains control bits that make it possible to individually set, clear, or toggle the logic state of a single pad by manipulating the output register.

### ESR Output Modification Register

#### OMR

#### ESR Output Modification Register

(00A8<sub>H</sub>)

System Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r														PCL1	PCL0
0															
r														PS1	PS0
w														w	w

Field	Bits	Type	Description
PSx (x=0-1)	x	w	<b>ESRx Pin Set Bit x</b> Setting this bit will set or toggle the corresponding bit in the output register OUT. Reading this bit returns 0.
PCLx (x=0-1)	x+16	w	<b>ESRx Pin Clear Bit x</b> Setting this bit will clear or toggle the corresponding bit in the port output register OUT. Reading this bit returns 0.
0	15:2, 31:18	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 256 Function of the Bits PCLx and PSx**

PCLx	PSx	Function
0	0	Bit OUT.Px is not changed
0	1	Bit OUT.Px is set
1	0	Bit OUT.Px is cleared
1	1	Bit OUT.Px is toggled

The logic level of a GPIO pin can be read via the read-only port input register IN. Reading the IN register always returns the current logical value at the GPIO pin independently whether the pin is selected as input or output.

## System Control Units (SCU)

### ESR Input Register

IN

#### ESR Input Register

(00AC<sub>H</sub>)System Reset Value: 0000 000X<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												P1	P0	rh rh	
r															

Field	Bits	Type	Description
Px (x=0-1)	x	rh	<b>Input Bit x</b> This bit indicates the level at the input pin ESRx. 0 <sub>B</sub> The input level of ESRx is 0 1 <sub>B</sub> The input level of ESRx is 1
0	31:2	r	<b>Reserved</b> Read as 0.

### Pad Disable Control Register

The pad structure of the AURIX™ TC3xx Platform GPIO lines offers the possibility to disable pad. This feature can be controlled by individual bits in the pad disable control register PDISC.

PDISC

#### Pad Disable Control Register

(018C<sub>H</sub>)System Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												PDIS1	PDIS0	rw rw	
r															

Field	Bits	Type	Description
PDISx (x=0-1)	x	rw	<b>Pad Disable for ESR Pin x</b> This bit disables the pad. 0 <sub>B</sub> Pad Px is enabled 1 <sub>B</sub> Pad Px is disabled
0	31:2	r	<b>Reserved</b> Read as 0; should be written with 0.

## System Control Units (SCU)

### ESR Pad Driver Mode Register

**PDR**

**ESR Pad Driver Mode Register**

(009C<sub>H</sub>)

**System Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								PL1	PD1	PL0	PD0				
r								rw	rw	rw	rw				

Field	Bits	Type	Description
<b>PD0</b>	1:0	rw	<b>Pad Driver Mode for ESR Pins 0</b> (See PDR register description in PORTs chapter)
<b>PL0</b>	3:2	rw	<b>Pad Level Selection for ESR Pins 0</b> (See PDR register description in PORTs chapter)
<b>PD1</b>	5:4	rw	<b>Pad Driver Mode for ESR Pins 1</b> (See PDR register description in PORTs chapter)
<b>PL1</b>	7:6	rw	<b>Pad Level Selection for ESR Pins 1</b> (See PDR register description in PORTs chapter)
<b>0</b>	31:8	r	<b>Reserved</b>

## System Control Units (SCU)

### 9.1.5 Boot Software Interface

In order to determine the correct starting point of operation for the software a minimum amount of hardware support is required. As much as possible is done via software. Some decisions have to be made in hardware because they must be known before any software is operational.

For a startup operation there are two general cases that have to be handled:

- Differentiation between Test Mode and Normal Mode for each Power-on Reset event (see [Section 9.1.5.1](#))
- Configuration of the boot option for each Application Reset event (see [Section 9.1.5.2](#))

#### 9.1.5.1 Configuration done with Start-up

At device power-on some basic operating mode selection has to be done. The first decision that has to be made is whether the device should operate in Test Mode or in Normal (Customer) Mode. The Test Mode is only for Infineon internal device testing, it is not intended for any customer and is not related to debug.

If the Normal Mode was selected the next decision is which debug interface type issued for debugging for this session (until the next power-on event).

**Table 257 Normal Mode / Test Mode Input Selection**

Field	Description
<b>TESTMODE</b>	<b>Latched TESTMODE Signal</b> 0 A Test Mode can be selected 1 Normal Mode is selected
<b>TRST</b>	<b>Latched TRST Signal</b> 0 The JTAG interface is active. 1 The DAP interface is active.

After these two decisions were made the detailed decision has to be made to define the real startup configuration. Most is made via the software and can be supported by some hardware selections depending on the startup configuration that should be selected.

#### 9.1.5.2 Start-up Configuration Options

The states of the HWCFG port pin inputs are latched on the rising edge of an Application Reset and stored in register STSTAT.HWCFG. The update of bit field STSTAT.HWCFG with the latched value is only done if bit STSTAT.LUDIS is cleared. If bit STSTAT.LUDIS is set then the value of STSTAT.HWCFG is not updated.

## System Control Units (SCU)

### 9.1.5.3 Boot Software Registers

#### 9.1.5.3.1 Start-up Status Registers

##### Start-up Status Register

###### STSTAT

###### Start-up Status Register

(00C0<sub>H</sub>)

PowerOn Reset Value: 000X 80XX<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							<b>RAMIN</b> T	0	0	0	<b>SPDE</b> N	<b>TRSTL</b>	0	<b>LUDIS</b>	<b>FCBAE</b>
r			r		rh	r	r	r	r	r	rh	rh	r	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>MODE</b>															<b>HWCFG</b>
rh															rh

Field	Bits	Type	Description
<b>HWCFG</b>	7:0	rh	<p><b>Hardware Configuration Setting</b>  This bit field contains the value that is used by the boot software.  This bit field is updated in case of an Application Reset with the content by register SWRSTCON.SWCFG if bit SWRSTCON.SWBOOT AND RSTSTAT.SW are set.  This bit field is updated in case of an ApplicationReset with the content of the latches of pins P14.2-P14.6, P10.5, P10.6 if bit SWRSTCON.SWBOOT OR RSTSTAT.SW are cleared and bit STSTAT.LUDIS is cleared.  This bit field is left unchanged in case of an ApplicationReset and is not updated with the content of the latches of pins P14.2-P14.6, P10.5, P10.6 if bit SWRSTCON.SWBOOT OR RSTSTAT.SW are cleared and bit STSTAT.LUDIS is set.</p> <p><i>Note:</i> The observed reset value after boot depends upon the state of the HWCFG pins</p>
<b>FTM</b>	14:8	rh	<p><b>Firmware Test Setting</b>  In Normal Mode this bit field is updated with 0000000<sub>B</sub> and should be ignored by the boot software.</p>
<b>MODE</b>	15	rh	<p><b>MODE</b>  This bit indicates if the Test Mode is entered or not.  0<sub>B</sub> A Test Mode can be selected  1<sub>B</sub> Normal Mode is selected</p>

**System Control Units (SCU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>FCBAE</b>	16	rh	<p><b>Flash Config. Sector Access Enable</b>            This bit can be cleared by setting bit STCON.CFCBAE.            This bit can be set by setting bit STCON.SFCBAE.</p> <p><i>Note:</i> Reset value of this bit is 0</p> <p>0<sub>B</sub> Flash config sector is not accessible. Instead the flash memory area is accessed.            1<sub>B</sub> Flash config sector is accessible. The flash memory area can not be accessed.</p>
<b>LUDIS</b>	17	rh	<p><b>Latch Update Disable</b>            This bit can be set by setting bit SYSCON.SETLUDIS.</p> <p><i>Note:</i> Reset value of this bit is 0</p> <p>0<sub>B</sub> Bit field STSTAT.HWCFG is automatically updated with the latched value of pins P14.2-P14.6, P10.5, P10.6            1<sub>B</sub> Bit field STSTAT.HWCFG is not updated with the latched value of pins P14.2-P14.6, P10.5, P1.6 P10.6</p>
<b>TRSTL</b>	19	rh	<p><b>TRSTL Status</b>            This bit simply displays the value of <u>TRST</u>.</p>
<b>SPDEN</b>	20	rh	<p><b>Single Pin DAP Mode Enable</b>            0<sub>B</sub> Single Pin DAP Mode is disabled            1<sub>B</sub> Single Pin DAP Mode is enabled</p>
<b>RAMINT</b>	24	rh	<p><b>RAM Content Security Integrity</b>            In normal operation this bit can be set or cleared by the application (via SYSCON.RAMINTM). If a test boot mode is entered, the bit is automatically cleared (and cannot be set again in test mode) because the content may have been altered</p> <p><i>Note:</i> This bit is reset only by a cold power-on reset.</p> <p>0<sub>B</sub> RAM Security Integrity cannot be guaranteed            1<sub>B</sub> RAM Security Integrity maintained</p>
<b>0</b>	18, 21, 22, 23, 27:25, 31:28	r	<p><b>Reserved</b>            Read as 0; should be written with 0.</p>

**System Control Units (SCU)****Start-up Configuration Register****STCON**
**Start-up Configuration Register (00C4<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>STP</b>	<b>CFCBAE</b>	<b>SFCBAE</b>													0
rwh	w	w													r

Field	Bits	Type	Description
<b>SFCBAE</b>	13	w	<b>Set Flash Config. Sector Access Enable</b> Setting this bit sets bit STSTAT.FCBAE. Reading this bit returns always a zero.  <i>Note:</i> If bits SFCBAE and CFCBAE are both set during the same access then bit STSTAT.FCBAE is set.
<b>CFCBAE</b>	14	w	<b>Clear Flash Config. Sector Access Enable</b> Setting this bit clears bit STSTAT.FCBAE. Reading this bit returns always a zero.
<b>STP</b>	15	rwh	<b>Start-up Protection Setting</b> This bit will be always set by FW and can't be reset. This bit is also cleared by an Application Reset. STP is automatically set when a shutdown trap occurs.  0 <sub>B</sub> Start-up code is executed. Start-up protection is disabled. 1 <sub>B</sub> Start-up code protection is active
<b>0</b>	12:0, 31:16	r	<b>Reserved</b> Read as 0; should be written with 0.

## System Control Units (SCU)

### 9.2 Trap Generation (TR)

The trap structure is shown in [Figure 69](#). A trap request trigger or the corresponding trap set bit (in register TRAPSET) can set flag bits in TRAPSTAT. Register bits in TRAPDIS[0:1] determine which CPUs shall receive traps from each TRAPSTAT trigger flag. By default after reset a trap is issued simultaneously to all CPUs. The TRAPSTAT trap flag can be cleared by software by writing to the corresponding bit in register TRAPCLR.

#### 9.2.1 Feature List

- CPU Traps may be triggered by transitions on ESRx pins or in response to a Safety Alarm
- CPU Trap trigger events are captured in a status register
- It is possible to generate or remove captured CPU Trap trigger events by software
- It is possible to disable or enable individual CPU Trap trigger events for individual CPUs

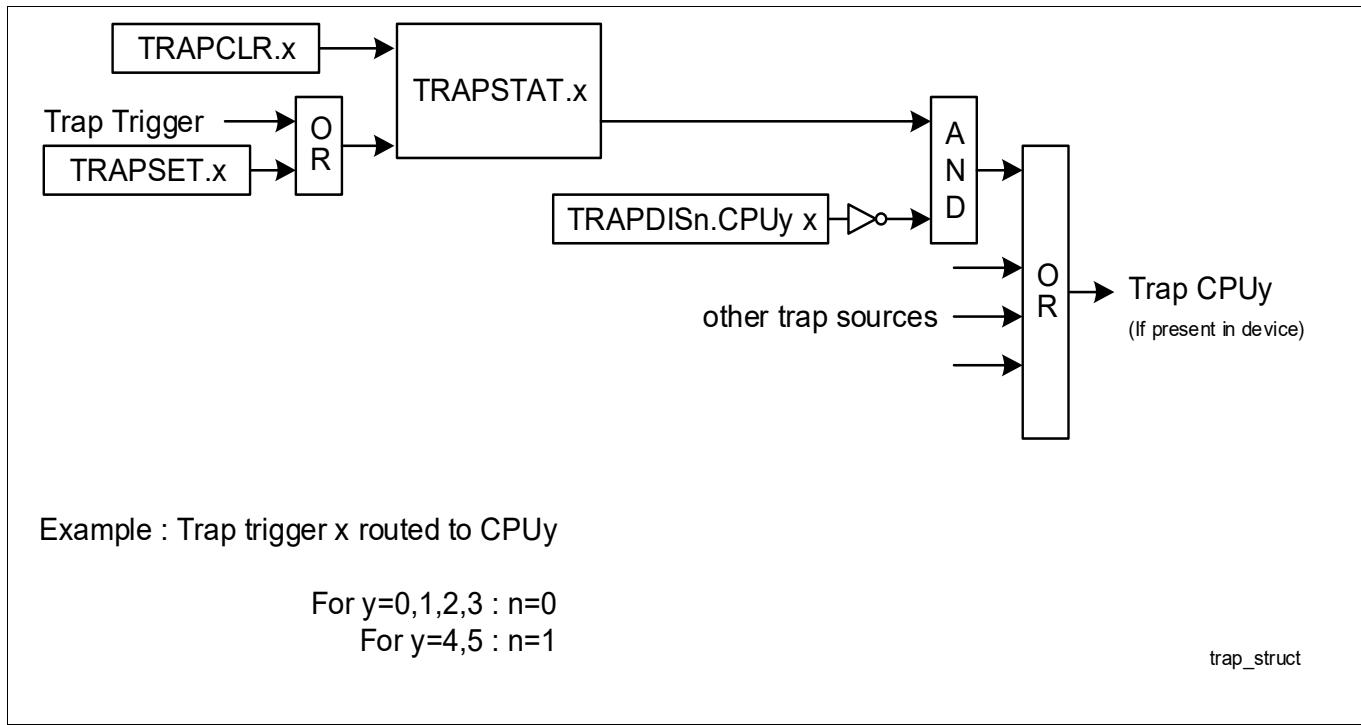
This module has the same basic features and functionality in all TC3xx devices although in some devices fewer CPUs may be available.

##### 9.2.1.1 Delta to AURIX

The TC3xx trap concept is based upon the trap concept of the TC2xx product range. The most significant change is:

- **TRAPDIS0** allows masking of trap types for individual CPUs (Similar **TRAPDIS1** added for additional CPUs)

#### 9.2.2 Trap Handling



**Figure 69 CPU Trap Generation**

As a trap is generated while the trap source is enabled AND the trap status flag is set, it is recommended to clear the trap status flag via TRAPCLR before a trap source is enabled in TRAPDISn. The trap status flag can be set

---

## System Control Units (SCU)

before the trap source is enabled and simply enabling the trap source can result in unintended CPU traps. At the end of a trap handling routine the trap status flag should be cleared.

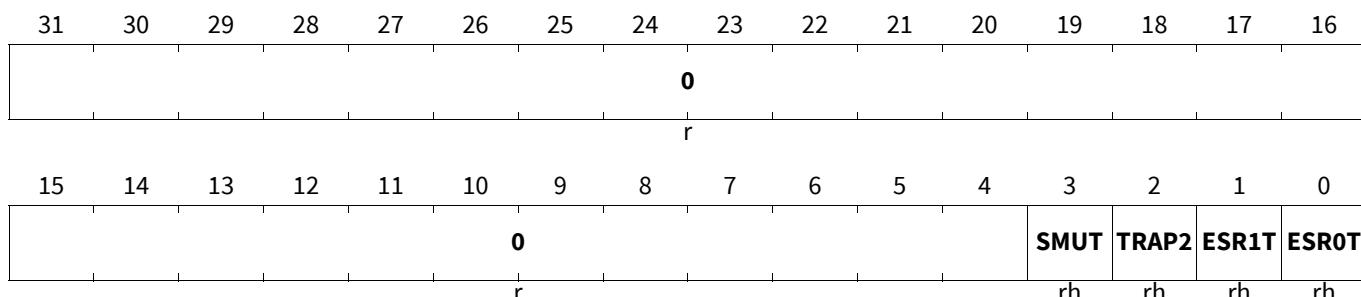
## System Control Units (SCU)

### 9.2.3 Trap Registers

#### Trap Status Register

##### TRAPSTAT

##### Trap Status Register

(0124<sub>H</sub>)System Reset Value: 0000 000X<sub>H</sub>

Field	Bits	Type	Description
<b>ESROT</b>	0	rh	<p><b>ESR0 Trap Request Flag</b></p> <p>This bit is set if an ESR0 event is triggered.</p> <p>This bit can be cleared by setting bit TRAPCLR.ESR0T.</p> <p>This bit can be set by setting bit TRAPSET.ESR0T.</p> <p><i>Note:</i> Observed reset value after boot will depend upon ARI mode because of ESR pin transition.</p> <p>0<sub>B</sub> No trap was requested since this bit was cleared the last time 1<sub>B</sub> A trap was requested since this bit was cleared the last time</p>
<b>ESR1T</b>	1	rh	<p><b>ESR1 Trap Request Flag</b></p> <p>This bit is set if an ESR1 event is triggered.</p> <p>This bit can be cleared by setting bit TRAPCLR.ESR1T.</p> <p>This bit can be set by setting bit TRAPSET.ESR1T.</p> <p><i>Note:</i> Reset value of this bit is 0</p> <p>0<sub>B</sub> No trap was requested since this bit was cleared the last time 1<sub>B</sub> A trap was requested since this bit was cleared the last time</p>
<b>TRAP2</b>	2	rh	<p><b>Trap Bit 2 Request Flag</b></p> <p>This bit can be cleared by setting bit TRAPCLR.TRAP2.</p> <p>This bit can be set by setting bit TRAPSET.TRAP2.</p> <p><i>Note:</i> Reset value of this bit is 0</p> <p>0<sub>B</sub> No trap was requested since this bit was cleared the last time 1<sub>B</sub> A trap was requested since this bit was cleared the last time</p>

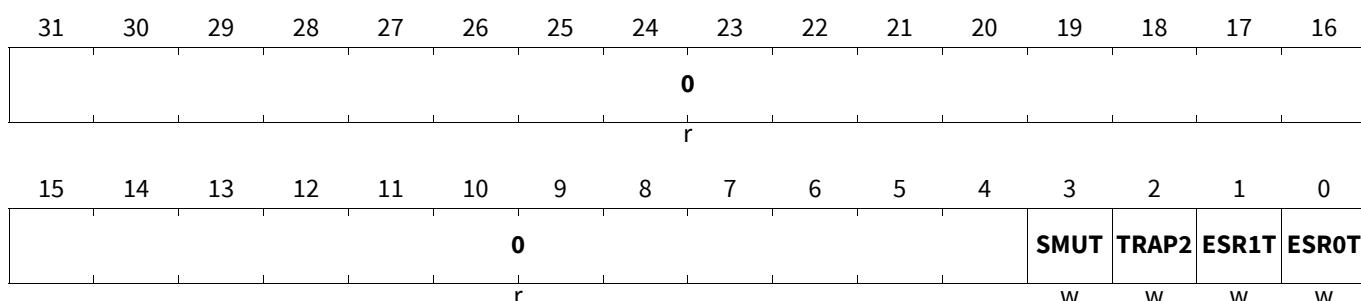
## System Control Units (SCU)

Field	Bits	Type	Description
<b>SMUT</b>	3	rh	<p><b>SMU Alarm Trap Request Flag</b>  This bit is set if an SMU Alarm is indicated.  This bit can be cleared by setting bit TRAPCLR.SMUT.  This bit can be set by setting bit TRAPSET.SMUT.</p> <p><i>Note:</i> Reset value of this bit is 0</p> <p>0<sub>B</sub> No trap was requested since this bit was cleared the last time  1<sub>B</sub> A trap was requested since this bit was cleared the last time</p>
<b>0</b>	31:4	r	<p><b>Reserved</b>  Read as 0.</p>

### Trap Set Register

#### TRAPSET

#### Trap Set Register

(0128<sub>H</sub>)System Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>ESR0T</b>	0	w	<p><b>Set Trap Request Flag ESR0T</b>  Setting this bit sets bit TRAPSTAT.ESR0T.  Clearing this bit has no effect.  Reading this bit returns always zero.</p>
<b>ESR1T</b>	1	w	<p><b>Set Trap Request Flag ESR1T</b>  Setting this bit sets bit TRAPSTAT.ESR1T.  Clearing this bit has no effect.  Reading this bit returns always zero.</p>
<b>TRAP2</b>	2	w	<p><b>Set Trap Request Flag TRAP2</b>  Setting this bit sets bit TRAPSTAT.TRAP2.  Clearing this bit has no effect.  Reading this bit returns always zero.</p>
<b>SMUT</b>	3	w	<p><b>Set Trap Request Flag SMUT</b>  Setting this bit sets bit TRAPSTAT.SMUT.  Clearing this bit has no effect.  Reading this bit returns always zero.</p>
<b>0</b>	31:4	r	<p><b>Reserved</b>  Read as 0; should be written with 0.</p>

## System Control Units (SCU)

### Trap Clear Register

#### TRAPCLR

##### Trap Clear Register

(012C<sub>H</sub>)

System Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															

Field	Bits	Type	Description
ESR0T	0	w	<b>Clear Trap Request Flag ESR0T</b> Setting this bit clears bit TRAPSTAT.ESR0T. Clearing this bit has no effect. Reading this bit returns always zero.
ESR1T	1	w	<b>Clear Trap Request Flag ESR1T</b> Setting this bit clears bit TRAPSTAT.ESR1T. Clearing this bit has no effect. Reading this bit returns always zero.
TRAP2	2	w	<b>Clear Trap Request Flag TRAP2</b> Setting this bit clears bit TRAPSTAT.TRAP2. Clearing this bit has no effect. Reading this bit returns always zero.
SMUT	3	w	<b>Clear Trap Request Flag SMUT</b> Setting this bit clears bit TRAPSTAT.SMUT. Clearing this bit has no effect. Reading this bit returns always zero.
0	31:4	r	<b>Reserved</b> Read as 0; should be written with 0.

### Trap Disable Register 0

#### TRAPDIS0

##### Trap Disable Register 0

(0130<sub>H</sub>)

Application Reset Value: FFFF FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1															
				CPU3S MUT	CPU3T RAP2T	CPU3E SR1T	CPU3E SR0T			1		CPU2S MUT	CPU2T RAP2T	CPU2E SR1T	CPU2E SR0T
				r	rw	rw	rw	r		rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1															
				CPU1S MUT	CPU1T RAP2T	CPU1E SR1T	CPU1E SR0T			1		CPU0S MUT	CPU0T RAP2T	CPU0E SR1T	CPU0E SR0T
				r	rw	rw	rw	r		rw	rw	rw	rw	rw	rw

## System Control Units (SCU)

Field	Bits	Type	Description
CPU0ESR0T	0	rw	<b>Disable Trap Request ESR0T on CPU0</b> 0 <sub>B</sub> A CPU0 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU0ESR1T	1	rw	<b>Disable Trap Request ESR1T on CPU0</b> 0 <sub>B</sub> A CPU0 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU0TRAP2T	2	rw	<b>Disable Trap Request TRAP2T on CPU0</b> 0 <sub>B</sub> A CPU0 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU0SMUT	3	rw	<b>Disable Trap Request SMUT on CPU0</b> 0 <sub>B</sub> A CPU0 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU1ESR0T	8	rw	<b>Disable Trap Request ESR0T on CPU1 (If product has CPU1)</b> 0 <sub>B</sub> A CPU1 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU1ESR1T	9	rw	<b>Disable Trap Request ESR1T on CPU1 (If product has CPU1)</b> 0 <sub>B</sub> A CPU1 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU1TRAP2T	10	rw	<b>Disable Trap Request TRAP2T on CPU1 (If product has CPU1)</b> 0 <sub>B</sub> A CPU1 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU1SMUT	11	rw	<b>Disable Trap Request SMUT on CPU1 (If product has CPU1)</b> 0 <sub>B</sub> A CPU1 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU2ESR0T	16	rw	<b>Disable Trap Request ESR0T on CPU2 (If product has CPU2)</b> 0 <sub>B</sub> A CPU2 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU2ESR1T	17	rw	<b>Disable Trap Request ESR1T on CPU2 (If product has CPU2)</b> 0 <sub>B</sub> A CPU2 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU2TRAP2T	18	rw	<b>Disable Trap Request TRAP2T on CPU2 (If product has CPU2)</b> 0 <sub>B</sub> A CPU2 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU2SMUT	19	rw	<b>Disable Trap Request SMUT on CPU2 (If product has CPU2)</b> 0 <sub>B</sub> A CPU2 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU3ESR0T	24	rw	<b>Disable Trap Request ESR0T on CPU3 (If product has CPU3)</b> 0 <sub>B</sub> A CPU3 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU3ESR1T	25	rw	<b>Disable Trap Request ESR1T on CPU3 (If product has CPU3)</b> 0 <sub>B</sub> A CPU3 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source

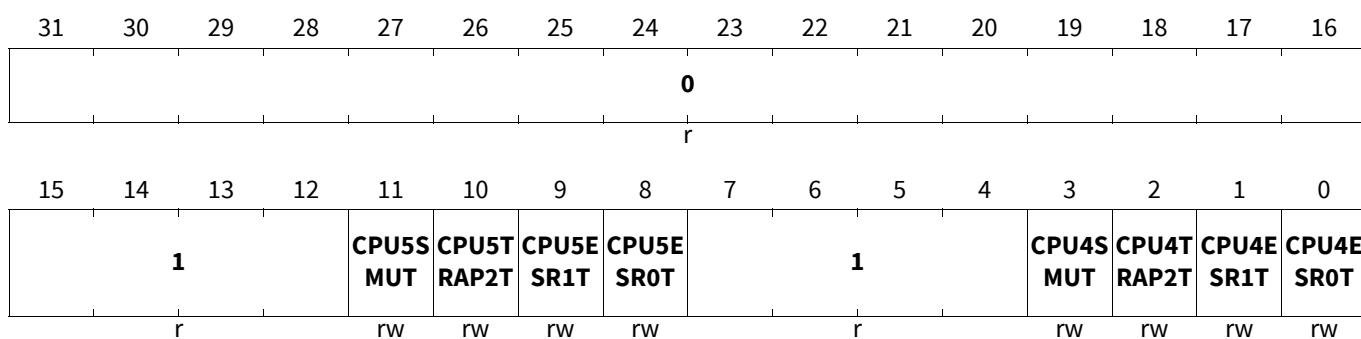
## System Control Units (SCU)

Field	Bits	Type	Description
CPU3TRAP2T	26	rw	<b>Disable Trap Request TRAP2T on CPU3 (If product has CPU3)</b> 0 <sub>B</sub> A CPU3 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU3SMUT	27	rw	<b>Disable Trap Request SMUT on CPU3 (If product has CPU3)</b> 0 <sub>B</sub> A CPU3 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
1	7:4, 15:12, 23:20, 31:28	r	<b>Reserved</b> Must only be written with one. Read as one.

### Trap Disable Register 1

#### TRAPDIS1

#### Trap Disable Register 1

(0120<sub>H</sub>)Application Reset Value: 0000 FFFF<sub>H</sub>

Field	Bits	Type	Description
CPU4ESR0T	0	rw	<b>Disable Trap Request ESR0T on CPU4 (If product has CPU4)</b> 0 <sub>B</sub> A CPU4 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU4ESR1T	1	rw	<b>Disable Trap Request ESR1T on CPU4 (If product has CPU4)</b> 0 <sub>B</sub> A CPU4 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU4TRAP2T	2	rw	<b>Disable Trap Request TRAP2T on CPU4 (If product has CPU4)</b> 0 <sub>B</sub> A CPU4 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU4SMUT	3	rw	<b>Disable Trap Request SMUT on CPU4 (If product has CPU4)</b> 0 <sub>B</sub> A CPU4 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
CPU5ESR0T	8	rw	<b>Disable Trap Request ESR0T on CPU5 (If product has CPU5)</b> 0 <sub>B</sub> A CPU5 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source

**System Control Units (SCU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>CPU5ESR1T</b>	9	rw	<b>Disable Trap Request ESR1T on CPU5 (If product has CPU5)</b>  0 <sub>B</sub> A CPU5 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
<b>CPU5TRAP2T</b>	10	rw	<b>Disable Trap Request TRAP2T on CPU5 (If product has CPU5)</b>  0 <sub>B</sub> A CPU5 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
<b>CPU5SMUT</b>	11	rw	<b>Disable Trap Request SMUT on CPU5 (If product has CPU5)</b>  0 <sub>B</sub> A CPU5 trap request can be generated for this source 1 <sub>B</sub> No trap request can be generated for this source
<b>1</b>	7:4, 15:12	r	<b>Reserved</b> Must only be written with one. Read as one.
<b>0</b>	31:16	r	<b>Reserved</b> Read as zero

## System Control Units (SCU)

### 9.3 System Register Unit (SRU)

The System Register Unit (SRU) contains miscellaneous control registers associated with various system functions.

#### 9.3.1 Feature List

This module contains control/status registers associated with various other functions/blocks:

- Lockstep control/status (for LCLx)
- LBIST control (for TCU)
- Overlay control (for CPUx)
- Start-up storage registers (for Boot SSW)
- Clock System control (for the Clock System configuration)

This module contains the same registers in all TC3xx products.

Some of the registers may be “reserved” in some products due to missing functionality or modules.

The functional groups of registers are as follows:

- CPU Lockstep Comparator Logic registers (LCL) (see [Section 9.3.2](#))
- Logic Built-in-Self-Test registers (LBIST) (see [Section 9.3.3](#))
- Clock System Control registers (see [Section 9.3.4](#))
- Overlay Control registers (OVC) (see [Section 9.3.5](#))
- Other registers (see [Section 9.3.6](#))

#### 9.3.1.1 Delta to AURIX

The most significant changes between the TC2xx SCU and TC3xx SRU are:

- Name change for clarification: SRU is a sub-module of the SCU cluster
- Structural repartitioning of SCU module with no impact on SW
- Added the LBIST description
- Some registers are now SE protected

### 9.3.2 Lockstep Comparator Logic Configuration

This section contains the registers which are used to configure and enable CPU Lockstep Mode. These registers are only writeable by start-up firmware and are configured by the BMI . See the Firmware and Lockstep Comparator Logic chapters for further details.

#### 9.3.2.1 Lockstep Comparator Logic Control Registers

##### LCL CPU0 and CPU2 Control Register

Provides control for CPU0and CPU2 Lockstep Comparator Logic blocks.

## System Control Units (SCU)

## LCLCON0

## LCL CPU0 and CPU2 Control Register

(0134<sub>H</sub>)

Reset Value: Table 258

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LSEN0</b>							<b>0</b>								<b>LS0</b>
rw							r								rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>LSEN2</b>							<b>0</b>								<b>LS2</b>
rw							r								rh

Field	Bits	Type	Description
<b>LS2</b>	0	rh	<b>Lockstep Mode Status</b> This bit indicates whether CPU2 is currently running in lockstep monitor mode (If product has lockstep capability on CPU2) $0_B$ Not in lockstep mode $1_B$ Running in lockstep mode
<b>LSEN2</b>	15	rw	<b>Lockstep Enable</b> This bit may only be written by SSW during boot. Enable lockstep CPU monitoring for the associated processor core, CPU2. If the product has no lockstep capability for CPU2, then this enables only the PFLASH access monitoring for CPU2. After cold reset, lockstep is enabled by default. The LSEN bit may be cleared during the boot to disable lockstep mode. SMU lockstep fault reporting should be disabled when lockstep is disabled. $0_B$ Lockstep is disabled $1_B$ Lockstep enabled (Default after Cold Power-On Reset)
<b>LS0</b>	16	rh	<b>Lockstep Mode Status</b> This bit indicates whether CPU0 is currently running in lockstep monitor mode $0_B$ Not in lockstep mode $1_B$ Running in lockstep mode
<b>LSEN0</b>	31	rw	<b>Lockstep Enable</b> This bit may only be written by SSW during boot. Enable lockstep CPU monitoring for the associated processor core, CPU0. After cold reset, lockstep is enabled by default. The LSEN bit may be cleared during the boot to disable lockstep mode. SMU lockstep fault reporting should be disabled when lockstep is disabled. $0_B$ Lockstep is disabled $1_B$ Lockstep enabled (Default after Cold Power-On Reset)
<b>0</b>	14:1, 30:17	r	<b>Reserved</b> will be read as $0_B$ , should be written as $0_B$

## System Control Units (SCU)

**Table 258 Reset Values of LCLCON0**

Reset Type	Reset Value	Note
Cold PowerOn Reset	8001 8001 <sub>H</sub>	

### LCL CPU1 and CPU3 Control Register

Provides control for CPU1 and CPU3 Lockstep Comparator Logic blocks.

#### LCLCON1

#### LCL CPU1 and CPU3 Control Register

(0138<sub>H</sub>)

Reset Value: [Table 259](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LSEN1</b>							<b>0</b>								<b>LS1</b>
rw							r								rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>LSEN3</b>							<b>0</b>								<b>LS3</b>
rw							r								rh

Field	Bits	Type	Description
<b>LS3</b>	0	rh	<p><b>Lockstep Mode Status</b>            This bit indicates whether CPU3 is currently running in lockstep monitor mode (If product has lockstep capability on CPU3)            0<sub>B</sub> Not in lockstep mode            1<sub>B</sub> Running in lockstep mode</p>
<b>LSEN3</b>	15	rw	<p><b>Lockstep Enable</b>            This bit may only be written by SSW during boot.            Enable lockstep CPU monitoring for the associated processor core, CPU3. If the product has no lockstep capability for CPU3, then this enables only the PFLASH access monitoring for CPU3.            After cold reset, lockstep is enabled by default. The LSEN bit may be cleared during the boot to disable lockstep mode. SMU lockstep fault reporting should be disabled when lockstep is disabled.            0<sub>B</sub> Lockstep is disabled            1<sub>B</sub> Lockstep enabled (Default after Cold Power-On Reset)</p>
<b>LS1</b>	16	rh	<p><b>Lockstep Mode Status</b>            This bit indicates whether CPU1 is currently running in lockstep monitor mode            0<sub>B</sub> Not in lockstep mode            1<sub>B</sub> Running in lockstep mode</p>

## System Control Units (SCU)

Field	Bits	Type	Description
LSEN1	31	rw	<p><b>Lockstep Enable</b>            This bit may only be written by SSW during boot.            Enable lockstep CPU monitoring for the associated processor core, CPU1. If the product has no lockstep capability for CPU1, then this enables only the PFLASH access monitoring for CPU1.            After cold reset, lockstep is enabled by default. The LSEN bit may be cleared during the boot to disable lockstep mode. SMU lockstep fault reporting should be disabled when lockstep is disabled.</p> <p>0<sub>B</sub> Lockstep is disabled            1<sub>B</sub> Lockstep enabled (Default after Cold Power-On Reset)</p>
0	14:1, 30:17	r	<p><b>Reserved</b>            will be read as 0<sub>B</sub>, should be written as 0<sub>B</sub></p>

**Table 259 Reset Values of LCLCON1**

Reset Type	Reset Value	Note
Cold PowerOn Reset	8001 8001 <sub>H</sub>	

### LCL Test Register

Provides the capability for software to inject a fault condition into the comparators of each Lockstep Comparator Logic block. The implementation should generate a single cycle fault each time the bit is written with '1'.

#### LCLTEST

<b>LCL Test Register</b>																<b>(013C<sub>H</sub>)</b>						<b>System Reset Value: 0000 0000<sub>H</sub></b>						
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	0	PLCLT5	PLCLT4	PLCLT3	PLCLT2	PLCLT1	PLCLT0	W	W	W	W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	0	LCLT5	LCLT4	LCLT3	LCLT2	LCLT1	LCLT0	W	W	W	W	W	W

Field	Bits	Type	Description
LCLT0	0	w	<p><b>LCL0 Lockstep Test</b>            Fault injection for LCL0. Reads as zero.</p> <p>0<sub>B</sub> No action            1<sub>B</sub> Inject single fault in LCL0</p>
LCLT1	1	w	<p><b>LCL1 Lockstep Test</b>            Fault injection for LCL1. Reads as zero.</p> <p>0<sub>B</sub> No action            1<sub>B</sub> Inject single fault in LCL1</p>

**System Control Units (SCU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>LCLT2</b>	2	w	<b>LCL2 Lockstep Test</b> Fault injection for LCL2. Reads as zero. $0_B$ No action $1_B$ Inject single fault in LCL2
<b>LCLT3</b>	3	w	<b>LCL3 Lockstep Test</b> Fault injection for LCL3. Reads as zero. $0_B$ No action $1_B$ Inject single fault in LCL3
<b>LCLT4</b>	4	w	<b>Pflash Lockstep Test fro CPU4</b> Fault injection for Pflash access lockstep of CPU4. Reads as zero. $0_B$ No action $1_B$ Inject single fault in PFLASH access lockstep
<b>LCLT5</b>	5	w	<b>Pflash Lockstep Test for CPU5</b> Fault injection for Pflash access lockstep of CPU5. Reads as zero. $0_B$ No action $1_B$ Inject single fault in PFLASH access lockstep
<b>PLCLT0</b>	16	w	<b>PFI0 Lockstep Test</b> Fault injection for PFI0 lockstep. Reads as zero. $0_B$ No action $1_B$ Inject single fault in PFI0 lockstep
<b>PLCLT1</b>	17	w	<b>PFI1 Lockstep Test</b> Fault injection for PFI1 lockstep. Reads as zero. $0_B$ No action $1_B$ Inject single fault in PFI1 lockstep
<b>PLCLT2</b>	18	w	<b>PFI2 Lockstep Test</b> Fault injection for PFI2 lockstep. Reads as zero. $0_B$ No action $1_B$ Inject single fault in PFI2 lockstep
<b>PLCLT3</b>	19	w	<b>PFI3 Lockstep Test</b> Fault injection for PFI3 lockstep. Reads as zero. $0_B$ No action $1_B$ Inject single fault in PFI3 lockstep
<b>PLCLT4</b>	20	w	<b>PFI4 Lockstep Test</b> Fault injection for PFI4 lockstep. Reads as zero. $0_B$ No action $1_B$ Inject single fault in PFI4 lockstep
<b>PLCLT5</b>	21	w	<b>PFI5 Lockstep Test</b> Fault injection for PFI5 lockstep. Reads as zero. $0_B$ No action $1_B$ Inject single fault in PFI5 lockstep
<b>0</b>	15:6, 31:22	r	<b>Reserved</b>

## System Control Units (SCU)

### 9.3.3 LBIST Support

AURIX™ TC3xx Platform supports automatic and user-triggered Logic Built-In Self-Test (LBIST) execution. Sub-sections below provide LBIST functional description and control registers for LBIST execution control and status.

#### 9.3.3.1 Introduction

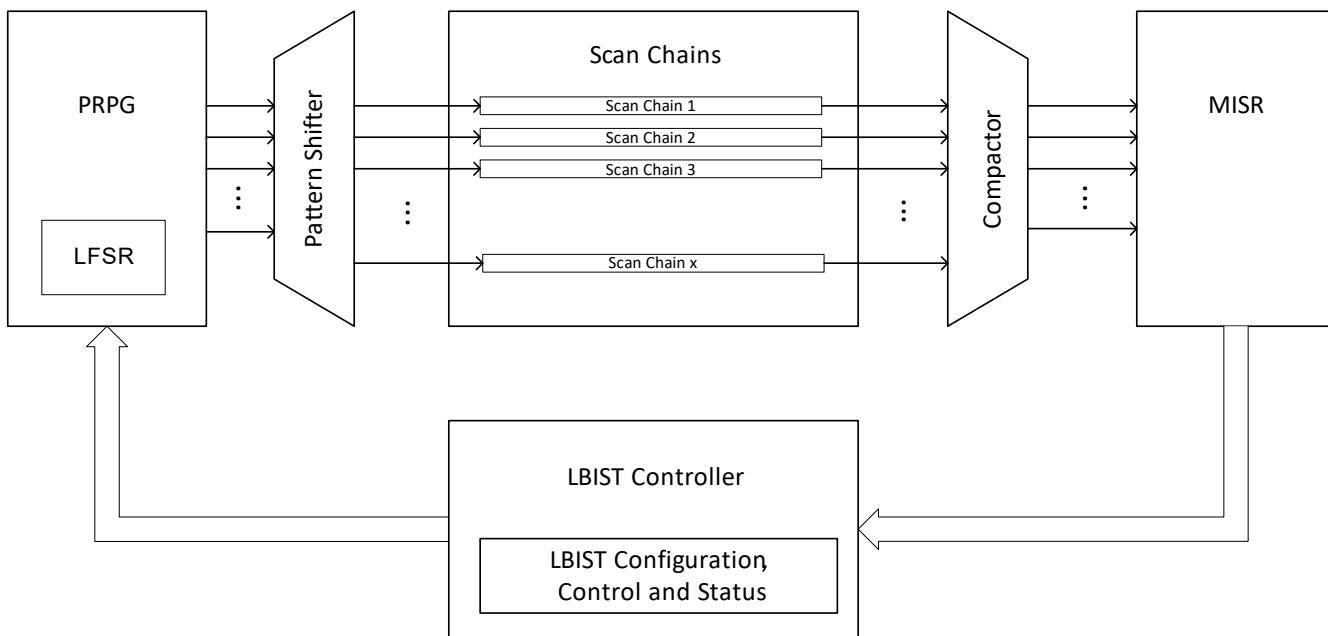
The LBIST is an on-chip hardware mechanism that can be used to detect MCU latent faults. The LBIST implementation in the AURIX TC3xx Platform allows to execute periodic self-tests for the MCU logic. The execution of the LBIST in MCU application mode is based on the DFT structures implemented for production testing and thus reusing scan chains, control and status mechanisms already available in MCU. There are two configurable ways to start LBIST execution: as a part of boot-up sequence or by application software in MCU functional mode. The results of LBIST execution are provided in LBIST result and status registers and can be used by application software to reach MCU safe state in case of detected latent faults.

#### 9.3.3.1.1 Functional Description

The Logic-BIST function is a structural test method for automatic in-system health-check of the digital design part. LBIST schemes use on-chip circuitry to generate test stimuli and analyze test responses, without any help from an chip-external test system. **Figure 70** depicts the TC3xx Platform LBIST architecture.

All the digital logic of the device with two exceptions is covered by LBIST. The two exceptions are:

- Logic in Power Management System, linked to the EVR and Standby Controller
- A part on the test control unit (including the LBIST controller)



**Figure 70 LBIST Architecture**

The LBIST structure applies pseudo-random patterns generated by a PRPG (Pseudo-Random Pattern Generator) to a full-scan circuit in parallel and compacts the test responses into a signature with a MISR (Multiple-Input Signature Register).

After LBIST execution has been successfully initiated from system-side (**LBISTCTRL0**, **LBISTCTRL1**, **LBISTCTRL2** registers), the whole design is switched into scan-mode. Afterwards the LBIST controller starts LBIST execution according to the configuration data (e.g. number of scan-loads, execution-speed or seed).

## System Control Units (SCU)

A Pseudo-Random Pattern Generator (PRPG) module is implemented to generate patterns to be shifted through scan-chains. The PRPG generates pseudo-random patterns using a Linear Feedback Shift Register (LFSR).

These patterns are loaded into the design through the scan-chains. In this case the core flip-flops are not operated in their functional mode but are organized in chains to allow a serial shift of these pseudo-random patterns into the complete MCU digital design. A specific shift counter keeps track of the number of shift-cycles to issue for each pattern.

The LBIST shift controller also uses the shift counter to separate the shift cycles from the capture cycles. Once a random pattern has been completely loaded into the digital logic scan-chains, the LBIST controller generates a single capture cycle.

In this case the flip-flops are switched back from serial scan-chain mode to their normal functional behavior, thus allowing to capture an actual state of the combinational logic.

The number of executed scan loads is selectable by software through the **LBISTCTRL0.PATTERNS** field. This field must be set before starting LBIST execution. The LBIST shift controller keeps track of how many patterns have been driven through the core and automatically stops the LBIST run when the programmed **LBISTCTRL0.PATTERNS** value has been reached.

During serial scan-chain shift cycles the outputs of the core flip-flop-chains are compressed through a XOR-gate network implemented in the Compactor module and permanently monitored through the MISR block, which generates a unique MISR signature. This signature allows application software to check if the MCU was tested by LBIST with or without errors.

The LBIST controller will automatically put the MISR into a static hold condition during loading of the first pseudo-random pattern into the scan-chains. This prevents the uninitialized values that are in the scan-chains prior to loading the first pattern from corrupting the MISR signature.

After LBIST execution is finished the MISR signature can be read out through the **LBISTCTRL3.SIGNATURE** field. Its value is only valid if **LBISTCTRL0.LBISTDONE** bit indicates a high value (i.e. LBIST run successfully terminated).

LBIST execution always terminates with an system warm reset, with an exception for the startup software that receives a cold reset (PMS/EVR settings of the startup software are not affected by this cold reset).

Therefore application software shall check if the LBIST execution was properly terminated and not interrupted by a PORST reset. The SCU.RSTSTAT register contains two status bits which capture LBIST termination status: RSTSTAT.LBPORST and RSTSTAT.LBTERM. The RSTSTAT.LBPORST status bit indicates if LBIST execution was terminated earlier due to a PORST assertion. If the status of this bit field is 0, the application must still check the RSTSTAT.LBTERM to check if the LBIST was terminated properly. The RSTSTAT.LBTERM stores the status if the LBIST execution was terminated properly.

A successfully finished LBIST procedure is indicated by the **LBISTCTRL0.LBISTDONE** bit. Value of **LBISTCTRL0.LBISTDONE** bit is not affected by the System or Application reset (it preserves its value). In case of warm or cold power-on reset, it resets LBISTDONE bit to 0, and soon after, if LBIST is configured to start, it will get its new result value.

The possible LBIST configuration options are located in the **LBISTCTRL0**, **LBISTCTRL1**, **LBISTCTRL2** registers.

The **LBISTCTRL0.PATTERNS** field defines the LBIST pattern count (i.e. number of scan-loads) which will be executed during LBIST procedure.

The value programmed to the **LBISTCTRL0.PATTERNS** field determines the number of scan-capture phases and not the number of scan-chain load/unload phases: a value of 0x00001 will result in two scan-chain loads with one capture in-between; a value of 0x00002 will result in 3 scan-chain loads with 2 captures, etc.). Consequently a value of 0x00000 is not valid, because no capture would be executed in this case.

## System Control Units (SCU)

The LBIST execution speed (i.e. shift-frequency) can be influenced through **LBISTCTRL1.LBISTFREQU** field.

In principle the LBIST Controller is operating on base of a 100MHz oscillator, whose output frequency can be scaled down by an integer value between 1 to 16 (determined through **LBISTCTRL1.LBISTFREQU** value).

A value of 3 or higher is recommended for a stable LBIST execution: 3=33MHz, 4=25MHz, 5=20MHz, ...,16=6,25MHz. Values 1 and 2 are not to be used for a stable LBIST: execution1=100MHz[not to be used], 2=50MHz[not to be used].

The generation of pseudo-random patterns by the PRPG can be influenced through the LBIST Seed option determined by **LBISTCTRL1.SEED** field.

This allows to vary the data-sequence, which is provided by the PRPG as input source for the core scan-chains.

In this way several LBIST runs with varying pattern sequences are possible. This allows to execute multiple time-sliced LBIST sequences with comparable test-coverage results as would be achieved in case of a single long running LBIST procedure.

The static GPIO behavior during LBIST-execution is selectable through the **LBISTCTRL1.BODY** bit.

Here the selection between tri-state (**LBISTCTRL1.BODY='1'**) and a weak pull-up (**LBISTCTRL1.BODY='0'**) behavior is possible.

As a rule all GPIO output drivers and input Schmitt-Triggers stay permanently disabled during LBIST-execution in order to isolate the device from the application-system.

The on-chip power consumption during LBIST-execution can be influenced through the **LBISTCTRL1.SPLITSH**-field. As a default all scan-chains are shifted concurrently during LBIST operation (**LBISTCTRL1.SPLITSH=0x0**), which can cause resulting power consumption exceeding the one of the maximum power pattern.

*Note:* Please check the LBIST power jump and execution time parameters with defined configurations in the device datasheet.

This might cause stability problems in certain application environments depending on the general supply strategy.

If power related stability problems should occur during LBIST execution it is possible to divide the core scan-chains either into two (**LBISTCTRL1.SPLITSH=0x5**) or four (**LBISTCTRL1.SPLITSH=0x4**) partitions, which are then shifted sequentially.

As a consequence the LBIST execution time will increase by a factor of 2 respectively a factor of 4.

To start a LBIST operation the **LBISTCTRL0.LBISTREQ** and **LBISTCTRL0.LBISTREQRED** bits must be written high. However a new LBIST sequence will only start if the **LBISTCTRL0.LBISTDONE** bit is reflecting a low value (i.e. no LBIST was executed since last power-on-reset).

The high value of **LBISTCTRL0.LBISTDONE** bit is an indication that the LBIST operation terminated normally and the SIGNATURE value is ready for readout.

Once this **LBISTCTRL0.LBISTDONE** bit is set to high, its value can be changed back to low by setting the **LBISTCTRL0.LBISTRES** bit to '1': this will reset **LBISTCTRL3.SIGNATURE** of the previous run and clear **LBISTCTRL0.LBISTDONE** status bit.

### LBIST Functional Safety Aspects

In order to fulfill ASIL-D safety standards the MCU must indicate any unintended states of LBIST-control block or of general test-mode enabling signals to the Application Software. For that purpose two alarm signals are implemented:

## System Control Units (SCU)

- LBIST-Alarm: This alarm signal shall be activated if the LBIST-FSM is NOT in the Reset/Idle state or if some global LBIST-enable signals are active. Consequently LBIST-alarm-signal is always active during LBIST execution.
- Test-Mode-Alarm: This alarm signal is activated if any of the TCU's general test-mode enabling signals are in unintended active state. Consequently Test-Mode-Alarm signal is always active if the MCU is operating in general test-mode.

Application software can trigger both listed above Alarms through error injection. For that purpose LBIST provides a single control bit in **LBISTCTRL0.LBISTERRINJ** to trigger LBIST-Alarm and Test-Mode-Alarm to SMU module.

To address unintended LBIST execution failure mode, the redundant LBIST request bit is implemented in the **LBISTCTRL0**. To start a LBIST operation the **LBISTCTRL0.LBISTREQ** and **LBISTCTRL0.LBISTREQRED** bits must be written high

### 9.3.3.2 LBIST Control Register

The LBISTCTRL Control Register provides the link between software and the LBIST-controller.

#### Logic BIST Control 0 Register

**LBISTCTRL0**

**Logic BIST Control 0 Register (0164<sub>H</sub>) Reset Value: Table 260**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LBIST REQR ED</b>	<b>LBIST ERRIN J</b>	0	<b>LBIST DONE</b>				0								<b>PATTERNS</b>
rw	rwh	r	rh				r								rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
														<b>LBIST RES</b>	<b>LBIST REQ</b>
														w	w

Field	Bits	Type	Description
<b>LBISTREQ</b>	0	w	<p><b>LBIST Request</b></p> <p>If written high this bit requests the execution of an automatic scan-test procedure. The request will only be approved if <b>LBISTCTRL0.LBISTDONE</b> bit reflects a '0'-value (i.e. no LBIST-procedure was triggered since the last power-on-reset or LBIST-controller has been restarted through the <b>LBISTCTRL0.LBISTRES</b>-bit). If read this bit always returns a '0'.</p> <p>This bit shall be implemented in a safety-relevant way to avoid unintended activation of LBIST during application.</p> <p><i>Note:</i> LBIST execution time depends on the number of scan-loads as defined in the PATTERNS field.</p>

## System Control Units (SCU)

Field	Bits	Type	Description
<b>LBISTRES</b>	1	w	<p><b>LBIST-Reset- LBISTRES</b> If written high this bit synchronously brings back the LBIST-controller to its initial Reset/Idle-state and also clears the stored MISR-signature to allow another execution from CPU-side. As a consequence the <b>LBISTCTRL0.LBISTDONE</b>- and SCU_LBISTCTRL3.SIGNATURE-bits will be set to '0'. If read this bit always returns a '0'.</p> <p><b>Note:</b> <i>It is strongly recommended to not change the LBISTFREQ parameter in LBISTCTRL1 after this bit has been set to '1', because there is no guarantee that the new frequency parameter value will be transferred to the LBIST-controller in-time before the next LBIST-run is started from user-side (i.e. LBISTCTRL0.LBISTREQ is set to '1').</i></p>
<b>PATTERNS</b>	19:2	rw	<p><b>LBIST Pattern Number</b> This field defines the number of scan-patterns (i.e. scan-loads), which will be executed during the LBIST-procedure. Please note that the value programmed to this field determines the number scan-capture phases not the number of scan-chain load/unload phases (i.e. a value of 0x00001 will result in two scan-chain loads with one capture in-between; a value of 0x00002 will result in 3 scan-chain loads with 2 captures, etc.). Consequently a value of 0x00000 is not valid, because no capture would be executed in this case.</p>
<b>LBISTDONE</b>	28	rh	<p><b>LBIST Execution Indicator</b> This bit indicates the actual LBIST-controller execution status:</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> No LBIST executed since last power-on-reset or LBIST-controller has been restarted (via <b>LBISTCTRL0.LBISTRES</b> function). Values in SCU_LBISTCTRL3.SIGNATURE-field are all set to '0'.</li> <li>1<sub>B</sub> At least one LBIST-procedure successfully finished since last power-on-reset. Values in SCU_LBISTCTRL3.SIGNATURE-field reflect the resulting MISR-signature.</li> </ul>
<b>LBISTERRINJ</b>	30	rwh	<p><b>LBIST / Test-Mode Alarm Error Injection</b> If written high this bit triggers both, the LBIST- and the test-mode-alarm. This is required to allow self-testing of all LBIST-(and test-mode-)related safety mechanisms in the TCU. The bit will be reset automatically once the LBIST and test-mode alarm indicator signals from TCU are asserted. From these indicator signals SCU will also generate corresponding alarm trigger signals for SMU.</p>
<b>LBISTREQRED</b>	31	rw	<p><b>LBIST Request Redundancy</b> This bit represents the safety double of LBISTCTRL0.LBISTREQ. In order to generate a new LBIST request both, LBISTREQRED and LBISTREQ bits must be set to high due to safety reasons. The request will only be approved if LBISTCTRL0.LBISTDONE bit reflects a '0'-value. If read this bit always returns a '0'.</p>
<b>0</b>	27:20, 29	r	<p><b>Reserved</b> Read as 0; should be written with 0.</p>

## System Control Units (SCU)

**Table 260** Reset Values of LBISTCTRL0

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
CFS Value	000– 0000 0000----- ----- ----- 00 <sub>B</sub>	LBISTDONE bit-field is not affected by system or application reset. It is reset to 0 after power-on reset and soon after LBIST is run it gets its new value. The correct CFS value of the PATTERNS bit-fields has to be looked up in the product-specific appendix document.

## Logic BIST Control 1 Register

LBISTCTRL1

## Logic BIST Control 1 Register

(0168<sub>H</sub>)

## **Reset Value: Table 261**

Field	Bits	Type	Description
<b>SEED</b>	18:0	rw	<p><b>LBIST Seed</b></p> <p>This field determines, which pattern is applied to the EDT-channel inputs 1-19 during LBIST execution.</p>
<b>SPLITSH</b>	26:24	rw	<p><b>LBIST Split-Shift Selection</b></p> <p>The value of this bit will allow to run LBIST with partitioned scan-shift operation in order to reduce the power consumption.</p> <p><math>000_B</math> Concurrent scan-shift is selected.  <math>\dots</math>  <math>011_B</math> Concurrent scan-shift is selected.  <math>100_B</math> Partitioned scan-shift is selected (four scan partitions).  <math>101_B</math> Partitioned scan-shift is selected (two scan partitions).  <math>110_B</math> Partitioned scan-shift is selected (four scan partitions).  <math>111_B</math> Partitioned scan-shift is selected (two scan partitions).</p>
<b>BODY</b>	27	rw	<p><b>Body Application Indicator</b></p> <p>The value of this bit will determine the static reset behavior of all GPIOs during LBIST execution. If set to low GPIOs will show a weak pull-up behavior, if set to high GPIOs are constrained to tri-state.</p> <p>A high value must be written to this bit in case LBIST shall be executed for body applications.</p>

## System Control Units (SCU)

Field	Bits	Type	Description
<b>LBISTFREQU</b>	31:28	rw	<p><b>LBIST Frequency Selection</b></p> <p>Through this register-field a pre-scaler factor between 1..16 is selectable for LBIST operation clock (derived from EVR-oscillator). This will allow to determine the LBIST scan-shift frequency. Value of these bits will be mirrored inside of LBIST-controller and become effective if a new LBIST-procedure has been successfully initiated from system-side (via <a href="#">LBISTCTRL0.LBISTREQ</a>).</p> <p><b>Note:</b> <i>It is strongly recommended not to change the value of this field after LBISTCTRL0.LBISTRES has been set to high, because there is no guarantee that the new frequency parameter value will be transferred to the LBIST-controller in-time before the next LBIST-run is started from user-side (i.e. LBISTCTRL0.LBISTREQ is set to '1')</i></p>
<b>0</b>	23:19	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 261 Reset Values of [LBISTCTRL1](#)**

Reset Type	Reset Value	Note
System Reset	0000 0000 <sub>H</sub>	
CFS Value	5400 0007 <sub>H</sub>	

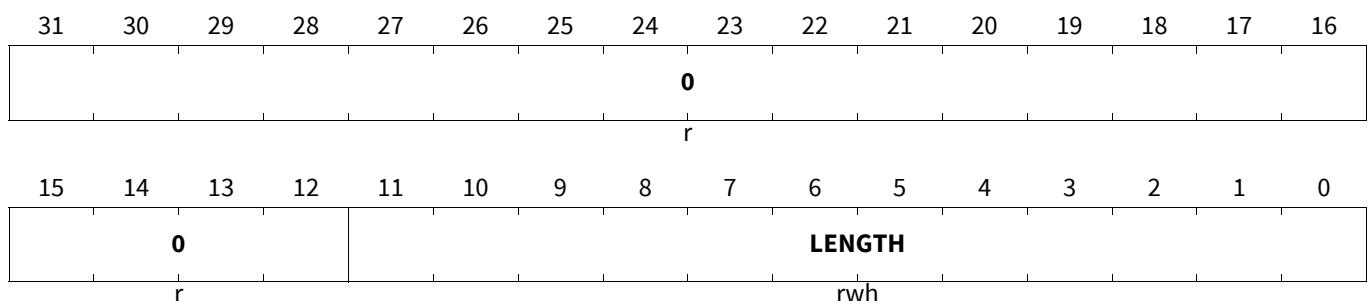
## Logic BIST Control 2 Register

### LBISTCTRL2

#### Logic BIST Control 2 Register

(016C<sub>H</sub>)

Reset Value: [Table 262](#)



Field	Bits	Type	Description
<b>LENGTH</b>	11:0	rwh	<p><b>LBIST Maximum Scan-Chain Length</b></p> <p>This field defines the number of shift-cycles for each LBIST scan-load. It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.</p>
<b>0</b>	31:12	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

## System Control Units (SCU)

**Table 262 Reset Values of LBISTCTRL2**

Reset Type	Reset Value	Note
System Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0086 <sub>H</sub>	

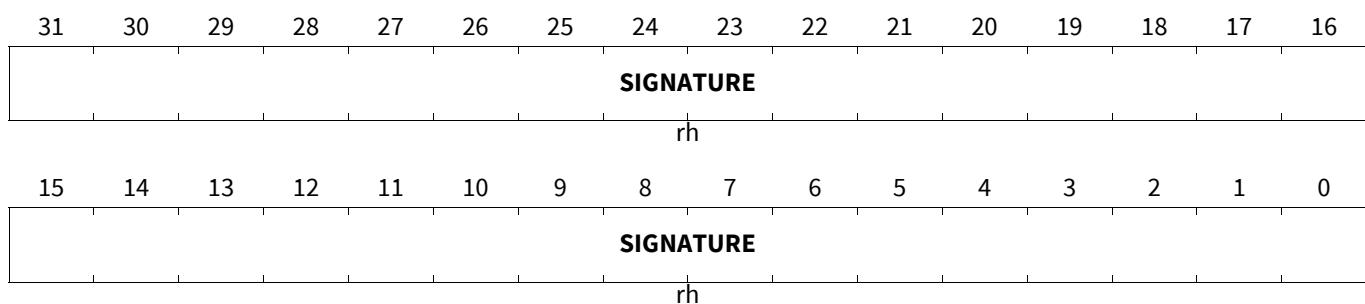
### Logic BIST Control 3 Register

**LBISTCTRL3**

**Logic BIST Control 3 Register**

(0170<sub>H</sub>)

**Reset Value: Table 263**



Field	Bits	Type	Description
<b>SIGNATURE</b>	31:0	rh	<b>LBIST Signature</b> This field reflects the MISR signature from the last LBIST execution. It is mirrored from LBIST-controller inside TCU and only valid if <b>LBISTCTRL0.LBISTDONE</b> is read with a high value. In case of a restart of the LBIST-controller (via <b>LBISTCTRL0.LBISTRES</b> function), the signature value will be synchronously reset to all-0. Please address the specific device appendix document for a description on the SIGNATURE value, depending on the LBIST run configuration.

**Table 263 Reset Values of LBISTCTRL3**

Reset Type	Reset Value	Note
System Reset	0000 0000 <sub>H</sub>	
CFS Value	0000 0000 <sub>H</sub>	

### 9.3.4 Clock System Control registers

The Clock System Control registers are implemented in the SRU and the SCU SPB bridge is used to access all of these registers. Nevertheless the register description is present in the specific Clock System chapter.

Please address the Clock System Chapter for a complete description of these registers.

## System Control Units (SCU)

### 9.3.5 Global Overlay Controls

The following registers control the Global Overlay functionality.

Overlay functionality is described in more detail in the CPU Overlay Subchapter.

Two registers globally control the overlay operation for all CPUs:

- Overlay Enable Register OVCENABLE disables or enables data access overlay individually for each CPU.
- Overlay Control Register OVCCON can be used to perform the following actions on a selected set of CPUs:
  - concurrently enable / disable selected overlay blocks,
  - concurrently disable overlay blocks,
  - invalidate data cache.

All overlay control registers are reset to their default values with the Application Reset . A special debug reset is not considered.

The external overlay feature is not available in product variants offering ADAS functionality.

#### 9.3.5.1 Global Overlay Control

##### Overlay Enable Register

###### OVCENABLE

###### Overlay Enable Register

(01E0<sub>H</sub>)

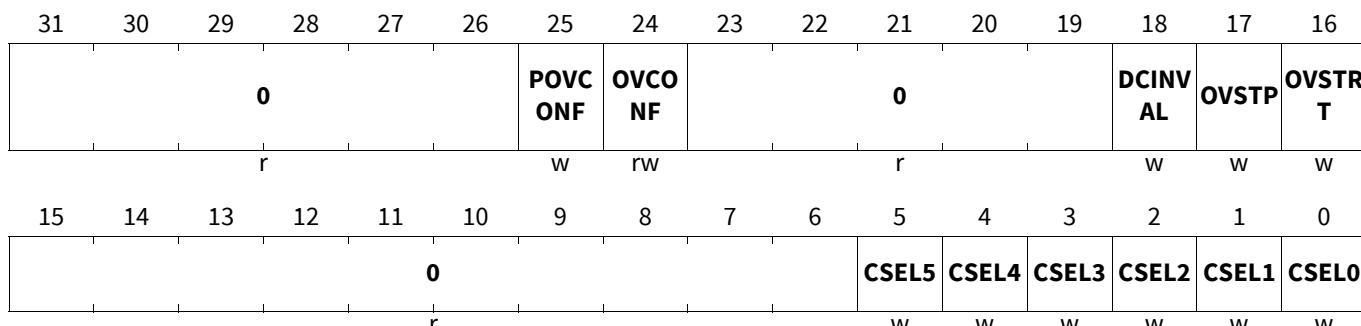
Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
0															
r															
0								OVEN5	OVEN4	OVEN3	OVEN2	OVEN1	OVENO		
r								rw	rw	rw	rw	rw	rw		

Field	Bits	Type	Description
OVEN0	0	rw	<b>Overlay Enable 0</b> 0 <sub>B</sub> OVC is disabled on CPU0. All Overlay redirections are disabled regardless of the state of OVC0_RABRY.OVEN. 1 <sub>B</sub> OVC is enabled on CPU0.
OVEN1	1	rw	<b>Overlay Enable 1 (If product has CPU1)</b> 0 <sub>B</sub> OVC is disabled on CPU1. All Overlay redirections are disabled regardless of the state of OVC1_RABRY.OVEN. 1 <sub>B</sub> OVC is enabled on CPU1.
OVEN2	2	rw	<b>Overlay Enable 2 (If product has CPU2)</b> 0 <sub>B</sub> OVC is disabled on CPU2. All Overlay redirections are disabled regardless of the state of OVC2_RABRY.OVEN. 1 <sub>B</sub> OVC is enabled on CPU2.

**System Control Units (SCU)**

Field	Bits	Type	Description
<b>OVEN3</b>	3	rw	<b>Overlay Enable 3 (If product has CPU3)</b>  0 <sub>B</sub> OVC is disabled on CPU3. All Overlay redirections are disabled regardless of the state of OVC3_RABRY.OVEN. 1 <sub>B</sub> OVC is enabled on CPU3.
<b>OVEN4</b>	4	rw	<b>Overlay Enable 4 (If product has CPU4)</b>  0 <sub>B</sub> OVC is disabled on CPU4. All Overlay redirections are disabled regardless of the state of OVC4_RABRY.OVEN. 1 <sub>B</sub> OVC is enabled on CPU4.
<b>OVEN5</b>	5	rw	<b>Overlay Enable 5 (If product has CPU5)</b>  0 <sub>B</sub> OVC is disabled on CPU5. All Overlay redirections are disabled regardless of the state of OVC5_RABRY.OVEN. 1 <sub>B</sub> OVC is enabled on CPU5.
<b>0</b>	31:6	r	<b>Reserved</b> Read/write 0.

**Overlay Control Register****OVCCON****Overlay Control Register**(01E4<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>CSEL0</b>	0	w	<b>CPU Select 0</b> Return 0 if read. 0 <sub>B</sub> CPU0 not affected, 1 <sub>B</sub> Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU0.
<b>CSEL1</b>	1	w	<b>CPU Select 1 (If product has CPU1)</b> Return 0 if read. 0 <sub>B</sub> CPU1 not affected, 1 <sub>B</sub> Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU1.

**System Control Units (SCU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>CSEL2</b>	2	w	<p><b>CPU Select 2 (If product has CPU2)</b></p> <p>Return 0 if read.</p> <p>0<sub>B</sub> CPU2 not affected, 1<sub>B</sub> Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU2.</p>
<b>CSEL3</b>	3	w	<p><b>CPU Select 3 (If product has CPU3)</b></p> <p>Return 0 if read.</p> <p>0<sub>B</sub> CPU3 not affected, 1<sub>B</sub> Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU3.</p>
<b>CSEL4</b>	4	w	<p><b>CPU Select 4 (If product has CPU4)</b></p> <p>Return 0 if read.</p> <p>0<sub>B</sub> CPU4 not affected, 1<sub>B</sub> Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU4.</p>
<b>CSEL5</b>	5	w	<p><b>CPU Select 5 (If product has CPU5)</b></p> <p>Return 0 if read.</p> <p>0<sub>B</sub> CPU5 not affected, 1<sub>B</sub> Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU5.</p>
<b>OVSTRT</b>	16	w	<p><b>Overlay Start</b></p> <p>CPUs which are not selected are not affected.</p> <p>No action is taken if OVSTP is also set.</p> <p>Return 0 if read.</p> <p>0<sub>B</sub> No action 1<sub>B</sub> For each CPU selected with CSEL, all the blocks selected with OVCx_OSEL will be activated. In the selected CPUs all the blocks deselected with OVCx_OSEL will be deactivated.</p>
<b>OVSTP</b>	17	w	<p><b>Overlay Stop</b></p> <p>CPUs which are not selected are not affected</p> <p>No action is taken if OVSTRT is also set.</p> <p>Return 0 if read.</p> <p>0<sub>B</sub> No action 1<sub>B</sub> For CPUs selected with CSEL, all the overlay blocks are deactivated. OVCx_RABRy.OVEN bits are cleared.</p>
<b>DCINVAL</b>	18	w	<p><b>Data Cache Invalidate</b></p> <p>No function in devices without data cache in CPU.</p> <p>Data Cache is affected only in the CPUs selected with CSEL.</p> <p>Return 0 if read.</p> <p>0<sub>B</sub> No action 1<sub>B</sub> Data Cache Lines in DMI are invalidated<sup>1)</sup></p>

## System Control Units (SCU)

Field	Bits	Type	Description
<b>OVCONF</b>	24	rw	<p><b>Overlay Configured</b></p> <p>Overlay configured status bit</p> <p>This bit may be used as handshake bit between a debug device (via JTAG interface and Cerberus) and CPU(s).</p> <p>0<sub>B</sub> Overlay is not configured or it has been already started</p> <p>1<sub>B</sub> Overlay block control registers are configured and ready for overlay start</p>
<b>POVCONF</b>	25	w	<p><b>Write Protection for OVCONF</b></p> <p>This bit enables OVCONF write during OVCCON write. Return 0 if read.</p> <p>0<sub>B</sub> OVCONF remains unchanged.</p> <p>1<sub>B</sub> OVCONF can be changed with write access to register OVCCON</p>
<b>0</b>	15:6, 23:19, 31:26	r	<p><b>Reserved</b></p> <p>Read/write 0.</p>

- 1) Dirty (modified) cache lines are not effected by this operation. If data cache contains modified data, it is not invalidated, and has to be written-back and invalidated by the user. Therefore, it is highly recommended to either: access overlaid data in read-only mode, or use only non-cached access.

## System Control Units (SCU)

### 9.3.6 Miscellaneous System Control

This section collects different registers that serve various system purposes.

#### 9.3.6.1 System Control Register

##### System Control Register

###### SYSCON

**System Control Register** **(007C<sub>H</sub>)** **System Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								DDC	Res	0	0	SETLU DIS	RAMINTM	0	CCTRIG0
rw								rw	rw	r	r	w	w	r	rw

Field	Bits	Type	Description
<b>CCTRIG0</b>	0	rw	<b>Capture Compare Trigger 0</b> This bit is used to trigger the Synchronous Start feature of the CCU6.
<b>RAMINTM</b>	3:2	w	<b>RAM Integrity Modify</b> 00 <sub>B</sub> No effect 01 <sub>B</sub> Set STSTAT.RAMINT (No effect in test mode) 10 <sub>B</sub> Clear STSTAT.RAMINT 11 <sub>B</sub> No effect
<b>SETLU<sub>DIS</sub></b>	4	w	<b>Set Latch Update Disable</b> Setting this bit sets bit STSTAT.LUDIS. Clearing this bit has no effect. This bit reads always as zero.
<b>Res</b>	7, 15:9	rw	<b>Reserved</b> Write only the read value
<b>DDC</b>	8	rw	<b>Disable DXCPL</b> 0 <sub>B</sub> DXCPL not disabled 1 <sub>B</sub> DXCPL disabled
<b>0</b>	1, 5, 6, 31:16	r	<b>Reserved</b> Read as 0

## System Control Units (SCU)

### 9.3.6.2 Identification Registers

#### Chip Identification Register

The CHIPID register can be used to determine the sales part number of the device (See Product Datasheet).

In general, the part number is of the form SAx-TC3yzab-ccFddd bc, where:

- CHIPID.CHID = Device Series Class (y)
- CHIPID.CHPK = Package Class (z)
- CHIPID.SEC = Feature Option (HSM enable, derived from b)
- CHIPID.EEA = Feature Option (Emulation part enable, derived from b)
- CHIPID.FSIZE = Flash size (derived from c)
- CHIPID.VART is used to differentiate any other marking options or special variants (e.g. Non-standard Temperature Range (x), Frequency (d) or Package Pitch (e) as defined in the Product Datasheet.

#### CHIPID

##### Chip Identification Register

(0140<sub>H</sub>)

Reset Value: [Table 264](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>SEC</b>	<b>VART</b>		<b>FSIZE</b>			<b>UCODE</b>						<b>EEA</b>			
rw	rw		rw			rw						rh			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>CHID</b>				<b>CHPK</b>			<b>CHTEC</b>	<b>CHREV</b>							
rw				rw			r	r							

Field	Bits	Type	Description
<b>CHREV</b>	5:0	r	<b>Chip Revision Number</b> This bit field indicates the revision number of the AURIX™ TC3xx Platform device. The value of this bit field is defined in the product Data Sheet. Bits [3:0] are used to indicate the steps. These updates can be done with any metal-fix or FW ROM change. Bits [5:4] define the major silicon design steps (A, B, C, D, ...). These bits can be changed only with a major redesign. 00 <sub>H</sub> A-step silicon ... 0F <sub>H</sub> A-step silicon 10 <sub>H</sub> B-step silicon ... 1F <sub>H</sub> B-step silicon 20 <sub>H</sub> Reserved ... 2F <sub>H</sub> Reserved

**System Control Units (SCU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>																																
<b>CHTEC</b>	7:6	r	<p><b>Chip Family</b></p> <p>These bits indicate the product family and are changed only with a redesign.</p> <table> <tr><td>00<sub>B</sub></td><td>Reserved</td></tr> <tr><td>01<sub>B</sub></td><td>SAx-TC2xx</td></tr> <tr><td>10<sub>B</sub></td><td>SAx-TC3xx</td></tr> <tr><td>11<sub>B</sub></td><td>Reserved</td></tr> </table>	00 <sub>B</sub>	Reserved	01 <sub>B</sub>	SAx-TC2xx	10 <sub>B</sub>	SAx-TC3xx	11 <sub>B</sub>	Reserved																								
00 <sub>B</sub>	Reserved																																		
01 <sub>B</sub>	SAx-TC2xx																																		
10 <sub>B</sub>	SAx-TC3xx																																		
11 <sub>B</sub>	Reserved																																		
<b>CHPK</b>	11:8	rw	<p><b>Chip Package</b></p> <p>These bits indicate the package.</p> <p>For further details refer to the Product Datasheet</p> <p>Use future variant codes for downconfigured silicon</p> <table> <tr><td>0<sub>H</sub></td><td>Bare Die</td></tr> <tr><td>1<sub>H</sub></td><td>Reserved</td></tr> <tr><td>2<sub>H</sub></td><td>TQFP80</td></tr> <tr><td>3<sub>H</sub></td><td>TQFP100</td></tr> <tr><td>4<sub>H</sub></td><td>TQFP144</td></tr> <tr><td>5<sub>H</sub></td><td>LQFP176</td></tr> <tr><td>6<sub>H</sub></td><td>BGA180</td></tr> <tr><td>7<sub>H</sub></td><td>LFBGA292</td></tr> <tr><td>8<sub>H</sub></td><td>Reserved</td></tr> <tr><td>9<sub>H</sub></td><td>LFBGA516</td></tr> <tr><td>A<sub>H</sub></td><td>BGA216</td></tr> <tr><td>B<sub>H</sub></td><td>Reserved</td></tr> <tr><td>...</td><td></td></tr> <tr><td>F<sub>H</sub></td><td>Reserved</td></tr> </table>	0 <sub>H</sub>	Bare Die	1 <sub>H</sub>	Reserved	2 <sub>H</sub>	TQFP80	3 <sub>H</sub>	TQFP100	4 <sub>H</sub>	TQFP144	5 <sub>H</sub>	LQFP176	6 <sub>H</sub>	BGA180	7 <sub>H</sub>	LFBGA292	8 <sub>H</sub>	Reserved	9 <sub>H</sub>	LFBGA516	A <sub>H</sub>	BGA216	B <sub>H</sub>	Reserved	...		F <sub>H</sub>	Reserved				
0 <sub>H</sub>	Bare Die																																		
1 <sub>H</sub>	Reserved																																		
2 <sub>H</sub>	TQFP80																																		
3 <sub>H</sub>	TQFP100																																		
4 <sub>H</sub>	TQFP144																																		
5 <sub>H</sub>	LQFP176																																		
6 <sub>H</sub>	BGA180																																		
7 <sub>H</sub>	LFBGA292																																		
8 <sub>H</sub>	Reserved																																		
9 <sub>H</sub>	LFBGA516																																		
A <sub>H</sub>	BGA216																																		
B <sub>H</sub>	Reserved																																		
...																																			
F <sub>H</sub>	Reserved																																		
<b>CHID</b>	15:12	rw	<p><b>Chip Product</b></p> <p>These bits indicate the base product.</p> <p>For further details refer to the Product Datasheet</p> <table> <tr><td>0<sub>H</sub></td><td>Reserved</td></tr> <tr><td>1<sub>H</sub></td><td>Reserved</td></tr> <tr><td>2<sub>H</sub></td><td>SAx-TC32xx</td></tr> <tr><td>3<sub>H</sub></td><td>SAx-TC33xx</td></tr> <tr><td>4<sub>H</sub></td><td>Reserved</td></tr> <tr><td>5<sub>H</sub></td><td>SAx-TC35xx</td></tr> <tr><td>6<sub>H</sub></td><td>SAx-TC36xx</td></tr> <tr><td>7<sub>H</sub></td><td>SAx-TC37xx</td></tr> <tr><td>8<sub>H</sub></td><td>SAx-TC38xx (based on TC38x silicon)</td></tr> <tr><td>9<sub>H</sub></td><td>SAx-TC39xx</td></tr> <tr><td>A<sub>H</sub></td><td>SAx-TC3Axx</td></tr> <tr><td>B<sub>H</sub></td><td>Reserved for future variants</td></tr> <tr><td>...</td><td></td></tr> <tr><td>D<sub>H</sub></td><td>Reserved for future variants</td></tr> <tr><td>E<sub>H</sub></td><td>SAx-TC3Exx</td></tr> <tr><td>F<sub>H</sub></td><td>Reserved for future variants</td></tr> </table>	0 <sub>H</sub>	Reserved	1 <sub>H</sub>	Reserved	2 <sub>H</sub>	SAx-TC32xx	3 <sub>H</sub>	SAx-TC33xx	4 <sub>H</sub>	Reserved	5 <sub>H</sub>	SAx-TC35xx	6 <sub>H</sub>	SAx-TC36xx	7 <sub>H</sub>	SAx-TC37xx	8 <sub>H</sub>	SAx-TC38xx (based on TC38x silicon)	9 <sub>H</sub>	SAx-TC39xx	A <sub>H</sub>	SAx-TC3Axx	B <sub>H</sub>	Reserved for future variants	...		D <sub>H</sub>	Reserved for future variants	E <sub>H</sub>	SAx-TC3Exx	F <sub>H</sub>	Reserved for future variants
0 <sub>H</sub>	Reserved																																		
1 <sub>H</sub>	Reserved																																		
2 <sub>H</sub>	SAx-TC32xx																																		
3 <sub>H</sub>	SAx-TC33xx																																		
4 <sub>H</sub>	Reserved																																		
5 <sub>H</sub>	SAx-TC35xx																																		
6 <sub>H</sub>	SAx-TC36xx																																		
7 <sub>H</sub>	SAx-TC37xx																																		
8 <sub>H</sub>	SAx-TC38xx (based on TC38x silicon)																																		
9 <sub>H</sub>	SAx-TC39xx																																		
A <sub>H</sub>	SAx-TC3Axx																																		
B <sub>H</sub>	Reserved for future variants																																		
...																																			
D <sub>H</sub>	Reserved for future variants																																		
E <sub>H</sub>	SAx-TC3Exx																																		
F <sub>H</sub>	Reserved for future variants																																		

**System Control Units (SCU)**

Field	Bits	Type	Description
<b>EEA</b>	16	rh	<b>Emulation or ADAS Extension Available</b> Indicates if the emulation or ADAS extension hardware is available or not. 0 <sub>B</sub> EEC is not available (SAK-TC3xxxU or SAK-TC3xxxP ) 1 <sub>B</sub> EEC is available (SAK-TC3xxxE or SAK-TC3xxxF )
<b>UCODE</b>	23:17	rw	<b>µCode Version</b> This bit field displays the Version X.Y of the flash µCode.
<b>FSIZE</b>	27:24	rw	<b>Program Flash Size</b> This bit field indicates available program flash size for this device. For more details see Product Datasheet. 0 <sub>H</sub> 256 KByte Program Flash (SAx-TC3xxx-4Fx) 1 <sub>H</sub> 0.5 MByte Program Flash (SAx-TC3xxx-8Fx) 2 <sub>H</sub> 1.0 MByte Program Flash (SAx-TC3xxx-16Fx) 3 <sub>H</sub> 1.5 MByte Program Flash (SAx-TC3xxx-24Fx) 4 <sub>H</sub> 2.0 MByte Program Flash (SAx-TC3xxx-32Fx) 5 <sub>H</sub> 2.5 MByte Program Flash (SAx-TC3xxx-40Fx) 6 <sub>H</sub> 3.0 MByte Program Flash (SAx-TC3xxx-48Fx) 7 <sub>H</sub> 4.0 MByte Program Flash (SAx-TC3xxx-64Fx) 8 <sub>H</sub> 5.0 MByte Program Flash (SAx-TC3xxx-80Fx) 9 <sub>H</sub> 6.0 MByte Program Flash (SAx-TC3xxx-96Fx) A <sub>H</sub> 7.0 MByte Program Flash (SAx-TC3xxx-112Fx) B <sub>H</sub> 8.0 MByte Program Flash (SAx-TC3xxx-128Fx) C <sub>H</sub> 10.0MByteProgram Flash (SAx-TC3xxx-160Fx) D <sub>H</sub> 12.0MByteProgram Flash (SAx-TC3xxx-192Fx) E <sub>H</sub> 14.0MByteProgram Flash (SAx-TC3xxx-224Fx) F <sub>H</sub> 16.0MByteProgram Flash (SAx-TC3xxx-256Fx)
<b>VART</b>	30:28	rw	<b>Variant</b> This bit field is used for variant identification. It is used to identify product variants with non-standard temperature profile, max frequency, package pitch or customer feature-sets. For coding details see Product Datasheet
<b>SEC</b>	31	rw	<b>Security Device Available</b> This bit field indicates whether the product has a Hardware Security Module 0 <sub>B</sub> No Hardware Security Module 1 <sub>B</sub> Hardware Security Module is available

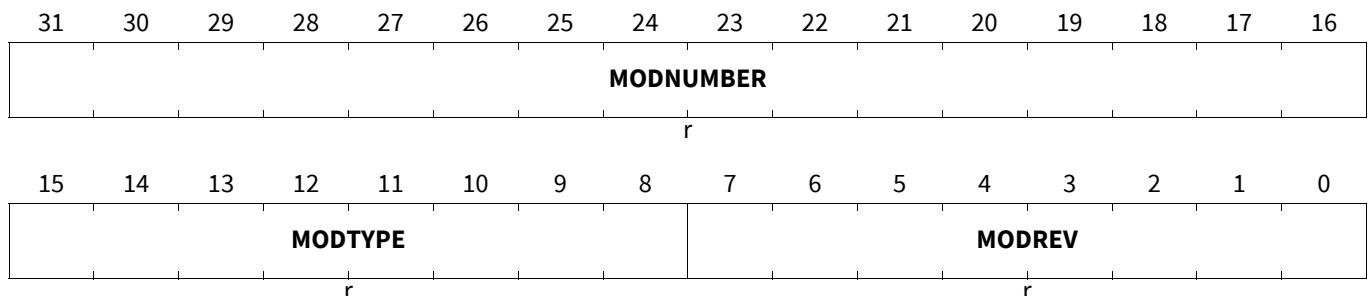
**Table 264 Reset Values of CHIPID**

Reset Type	Reset Value	Note
System Reset	XXXX XXXX <sub>H</sub>	

## System Control Units (SCU)

### Identification Register

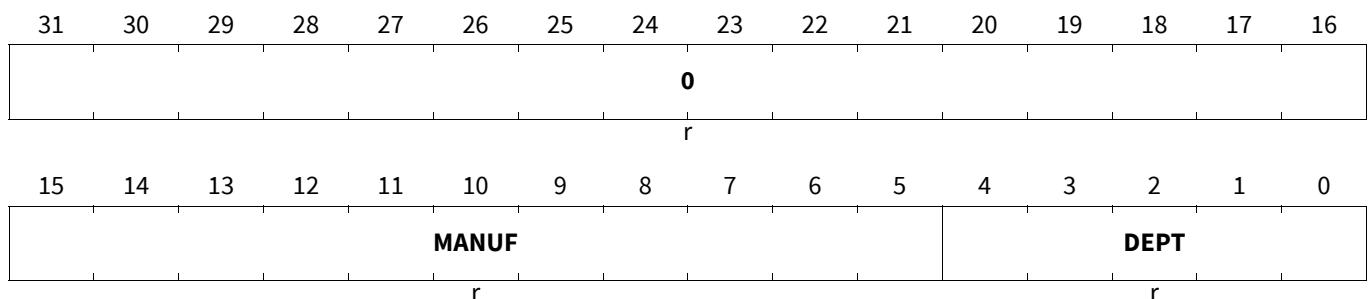
**ID**  
**Identification Register** **(0008<sub>H</sub>)** **System Reset Value: 00C4 C0C1<sub>H</sub>**



Field	Bits	Type	Description
<b>MODREV</b>	7:0	r	<b>Module Revision Number</b> This bit field indicates the revision number of the SCU module (C1 <sub>H</sub> ).
<b>MODTYPE</b>	15:8	r	<b>Module Type</b> This bit field is C0 <sub>H</sub> . It defines a 32-bit module
<b>MODNUMBER</b>	31:16	r	<b>Module Number</b> This bit field defines the module identification number. The identification number for the SCU is 00C4 <sub>H</sub>

### Manufacturer Identification Register

**MANID**  
**Manufacturer Identification Register** **(0144<sub>H</sub>)** **System Reset Value: 0000 1820<sub>H</sub>**



Field	Bits	Type	Description
<b>DEPT</b>	4:0	r	<b>Department Identification Number</b> = 00 <sub>H</sub> : indicates the Automotive & Industrial microcontroller department within Infineon Technologies.
<b>MANUF</b>	15:5	r	<b>Manufacturer Identification Number</b> This is a JEDEC normalized manufacturer code. MANUF = C1 <sub>H</sub> stands for Infineon Technologies.
<b>0</b>	31:16	r	<b>Reserved</b> Read as 0.

## System Control Units (SCU)

### 9.3.6.3 Start-up Software Memory Registers

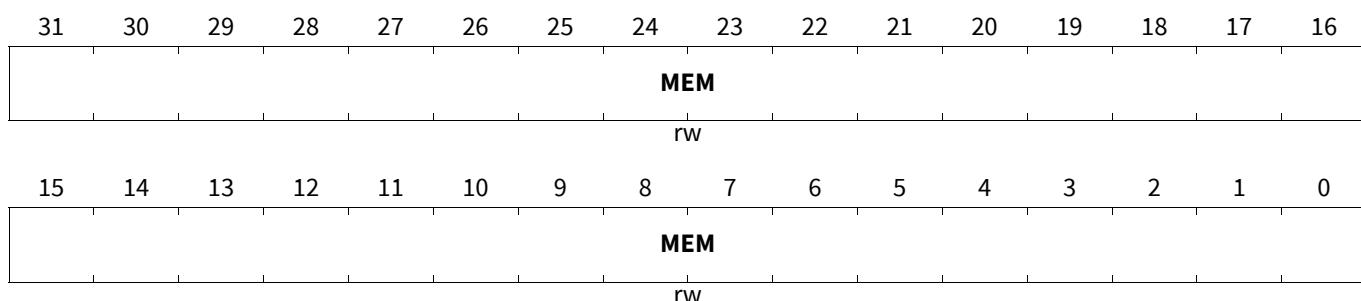
In this section one can see the address information about the STMEMx registers. These registers are used by the System Firmware for boot control. The description of these registers is available on the System Firmware specification chapter.

#### Start-up Memory Register 1

Please check this register description in the System Firmware chapter

##### STMEM1

**Start-up Memory Register 1** **(0184<sub>H</sub>)** **PowerOn Reset Value: 0000 0000<sub>H</sub>**



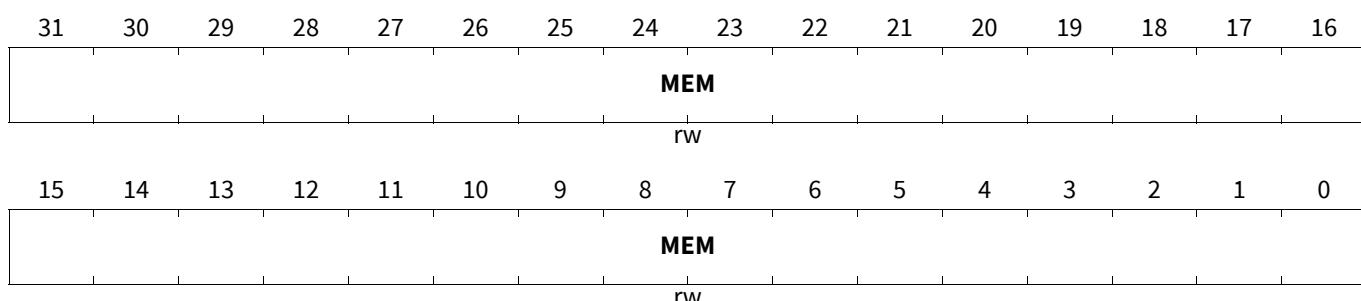
Field	Bits	Type	Description
MEM	31:0	rw	<b>Memory</b> This register is used by the start-up software as memory.

#### Start-up Memory Register 2

Please check this register description in the System Firmware chapter

##### STMEM2

**Start-up Memory Register 2** **(0188<sub>H</sub>)** **System Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
MEM	31:0	rw	<b>Memory</b> This register is used by the start-up software as memory.

#### Start-up Memory Register 3

Please check this register description in the System Firmware chapter

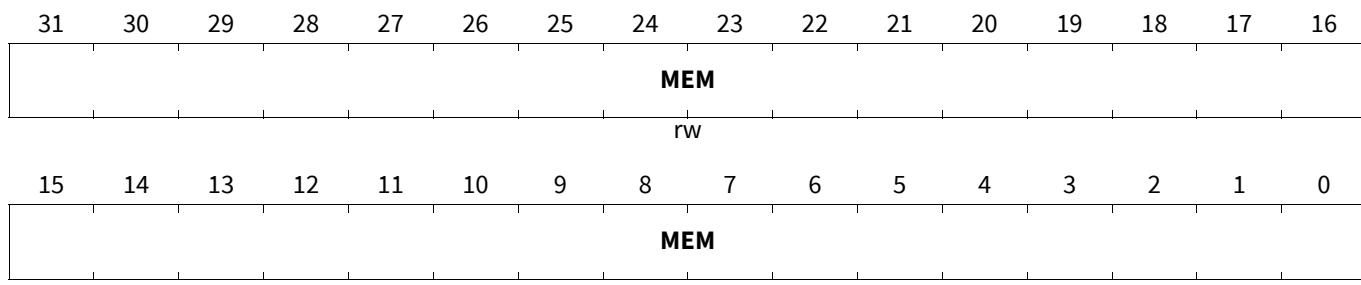
## System Control Units (SCU)

### STMEM3

#### Start-up Memory Register 3

(01C0<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
MEM	31:0	rw	<b>Memory</b> This register is used by the start-up software as memory.

#### Start-up Memory Register 4

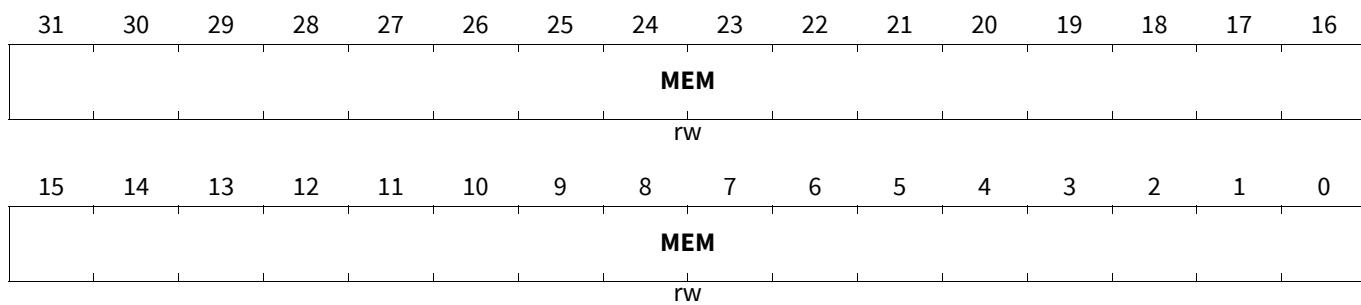
Please check this register description in the System Firmware chapter

### STMEM4

#### Start-up Memory Register 4

(01C4<sub>H</sub>)

Cold PowerOn Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
MEM	31:0	rw	<b>Memory</b> This register is used by the start-up software as memory.

#### Start-up Memory Register 5

Please check this register description in the System Firmware chapter

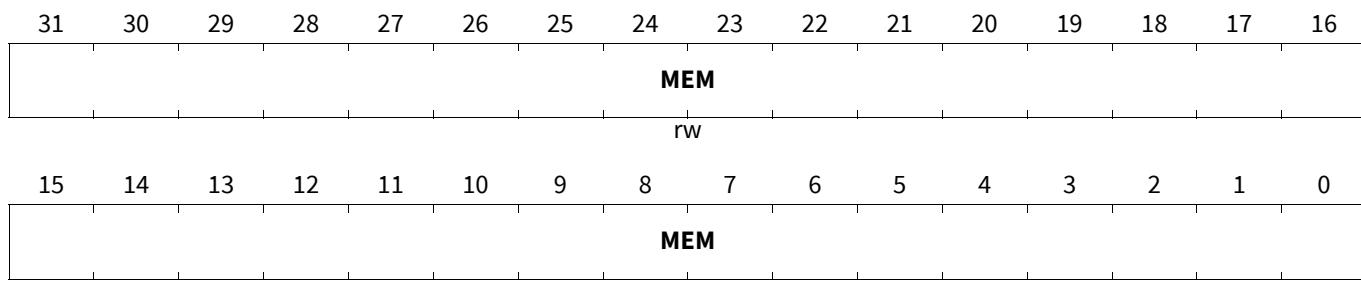
## System Control Units (SCU)

### STMEM5

#### Start-up Memory Register 5

(01C8<sub>H</sub>)

PowerOn Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
MEM	31:0	rw	<b>Memory</b> This register is used by the start-up software as memory.

#### Start-up Memory Register 6

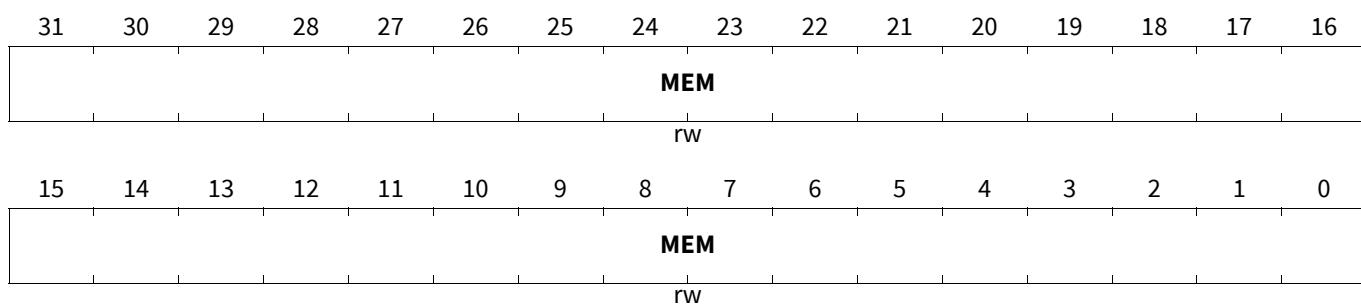
Please check this register description in the System Firmware chapter

### STMEM6

#### Start-up Memory Register 6

(01CC<sub>H</sub>)

System Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
MEM	31:0	rw	<b>Memory</b> This register is used by the start-up software as memory.

## System Control Units (SCU)

### 9.3.6.4 SCU Access Restriction Registers

#### Access Enable Register 00

The Access Enable Register 0 restricts write access to SCU, RCU, CCU and PMC registers marked “P0” so that they may only be written by specified bus masters (eg CPUs). See the Bus chapter for the mapping of TAG ID to specific system masters and CPUs).

#### ACCEN00

##### Access Enable Register 00 (03FC<sub>H</sub>) Application Reset Value: FFFF FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

Field	Bits	Type	Description
ENn (n=0-31)	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the SCU kernel addresses for transactions with the Master TAG ID n 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

#### Access Enable Register 01

#### ACCEN01

##### Access Enable Register 01 (03F8<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								0							
								r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							0								
							r								

Field	Bits	Type	Description
0	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

#### Access Enable Register 10

The Access Enable Register 1 restricts write access to SCU, RCU, CCU and PMC registers marked “P1” so that they may only be written by specified bus masters (eg CPUs). See the Bus chapter for the mapping of TAG ID to specific system masters and CPUs).

**System Control Units (SCU)****ACCEN10****Access Enable Register 10**(03F4<sub>H</sub>)**Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENn (n=0-31)	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the SCU kernel addresses for transactions with the Master TAG ID n 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

**Access Enable Register 11****ACCEN11****Access Enable Register 11**(03F0<sub>H</sub>)**Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								0							
								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							0								
							r								

Field	Bits	Type	Description
0	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

## System Control Units (SCU)

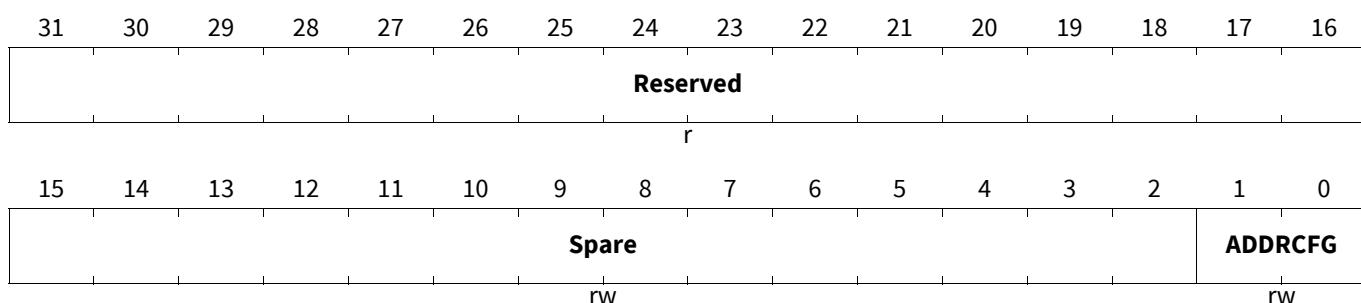
### 9.3.6.5 SOTA Address Map Control

#### Address Map Control Register

Provides the capability for firmware to install the currently used address map - to support SOTA

#### SWAPCTRL

**Address Map Control Register** **(014C<sub>H</sub>)** **System Reset Value: 0000 0001<sub>H</sub>**



Field	Bits	Type	Description
<b>ADDRCFG</b>	1:0	rw	<b>Address Configuration</b> Configures the currently used address map (standard/alternate selection) 00 <sub>B</sub> Reserved (00b shall not be written - if 00b is written an alarm at system level is triggered) 01 <sub>B</sub> Standard Address map active <b>10<sub>B</sub> Alternate Address map active</b> 11 <sub>B</sub> Reserved (11b shall not be written - if 11b is written an alarm at system level is triggered)
<b>Spare</b>	15:2	rw	<b>Spare address configuration registers</b> Spare read/write bits
<b>Reserved</b>	31:16	r	<b>Reserved</b> Read returns 0

## 9.4 Watchdog Timers (WDT)

### 9.4.1 Feature List

The TC3xx contains the following Watchdog Timers:

- One Safety Watchdog Timer
- One Watchdog Timer per CPU

Each Watchdog Timer has the following basic functionality:

- Programmable timebase and reload value
- Programmable password protection with configurable automatic password sequencing
- Programmable timestamp checking with programmable window
- Invalid or missing timer refresh sequence leads to Safety Alarm

## System Control Units (SCU)

- Possible to suspend the Watchdog operation during debug
- Critical register write-protection which can be unlocked only for short time-out duration

### 9.4.1.1 Changes to AURIX TM Family

The most significant changes between the AURIX TC2xx SCU WDT and AURIX TM TC3xx SCU WDT are:

- Register address changed
- Additional Watchdog Timers for additional CPUs
- ENDINIT unlock possible via new ENDINIT Timeout Counter (EICON registers) without affecting any CPU Watchdog Timer
- Safety ENDINIT unlock possible via new Safety ENDINIT Timeout Counter (SEICON registers) without affecting Safety Watchdog Timer
- Separate ACCEN protection range for Safety Watchdog Timer
- External WDT “Alive Heartbeat” Indication feature removed

### 9.4.1.2 Changes from TC39x A-Step to AURIX TC3xx

The following differences exist between the TC39x A-Step device and later TC3xx devices:

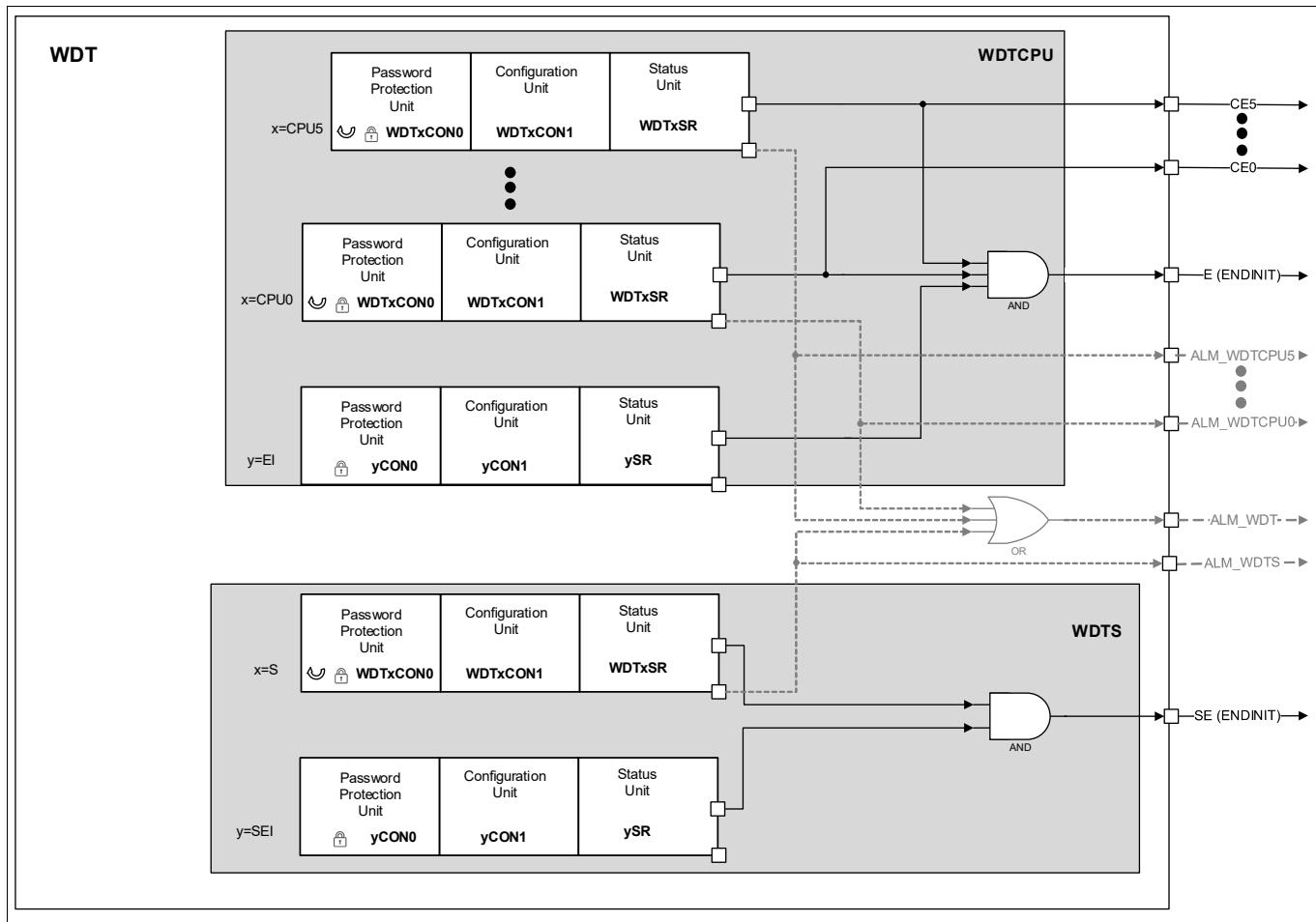
- In TC39X A-Step OCDS debug mode is not available for WDTS, EICON and SECICON.
- In later TC3xx devices the AURIX TC2XX solution is used for Suspend Mode Support. Dependency of WTDS/EICON/SEICON State is implemented according to [Table 270](#).

*Note: For WDTCPUy (y=CPU number), the Suspend Mode Support solution of AURIX TC2xx, TC39x A-Step and AURIX TM TC3xx SCU WDT are compatible ( see. [Table 269](#) ).*

### 9.4.2 Watchdog Timers Overview

The WDTs provide a highly reliable and secure way to detect and recover from software or hardware failure. The WDTs help to abort an accidental malfunction of a CPU or system within a user-specified time period.

## System Control Units (SCU)



**Figure 71 WDT Block Diagram**

In addition to this standard “Watchdog” function, each of the WDTs incorporates an End-of-Initialization (ENDINIT) feature which can protect critical registers from accidental writes.

Servicing the Watchdogs and modifying the ENDINIT bits are critical functions that must not be allowed in case of a system malfunction. To protect these functions a sophisticated scheme is implemented that requires a password and guard bits during accesses to the WDT control registers. Any write access that does not deliver the correct password or the correct value for the guard bits is regarded as a malfunction of the system, and results in a watchdog alarm. In addition, even after a valid access has been performed and an ENDINIT bit has been cleared to provide access to the critical registers, the Watchdog imposes a time limit for this access window. If the ENDINIT bit has not been properly set again before this limit expires, the system is assumed to have malfunctioned. These stringent requirements, although not guaranteed, nonetheless provide a high degree of assurance of the robustness of system operation.

Configuration options are available which enable a Watchdog service to additionally check code execution sequence and intermediate code execution time. If these checks are enabled then any incorrect sequence or out-of-limit execution time will also result in an SMU alarm request.

Expiry of any of the WDTs leads to an SMU alarm. It is possible to program the SMU to provide an interrupt or trap to provide some time for recovery or status recording before further action is taken (e.g. reset of the device or the CPU).

## System Control Units (SCU)

### 9.4.2.1 Safety Watchdog

The Safety Watchdog Timer provides an overall system level watchdog which is independent from the CPU Watchdog Timers and also provides another protection against unintended writes to safety critical system registers. When the Safety WDT is enabled, it can cause an SMU alarm request if it is not serviced within a user-programmable time period. A CPU must service the Safety WDT within this time interval to prevent this. The response to a Safety WDT timeout is configurable within the SMU. Hence, periodic servicing of the Safety WDT confirms that the system is functioning as expected.

Typically the SCU write protection (ACCEN) will be configured so that only restricted “Safety” CPU(s) can configure safety critical functionality. This includes the ability to service the Safety Watchdog. In addition, Safety Watchdog Timer disable/enable/configuration function requires a Safety ENDINIT password.

The registers marked “SE” in the SCU register overview table and other module Register Tables are considered to be static properties of a safety-critical system and are all write-protected after initialization. This write protection may be temporarily removed if the Safety ENDINIT bit is cleared.

### 9.4.2.2 CPU Watchdogs

The individual CPU Watchdog Timers provide the ability to monitor separate CPU execution threads without the need for software to co-ordinate the shared use of a common watchdog.

When a CPU WDT is enabled it can cause an SMU alarm request if it is not correctly serviced within a user-programmable time period. The CPU must service its CPU WDT within this time interval to prevent this. The response to each CPUy watchdog timeout is configurable within the SMU. Hence, periodic servicing of a CPU WDT confirms that the corresponding CPU is executing a software sequence as expected.

After a reset, CPU0 runs and CPU0 Watchdog Timer starts automatically.

Other CPUs are initially in a HALT state and therefore their corresponding Watchdog Timers are disabled. The other CPU Watchdog Timers are not configured to generate a time-out reset by default, but this can be enabled. A CPU Watchdog may only be configured, enabled or disabled by its corresponding CPU.

The registers marked “CEy” (y= CPU number) in SCU register overview table and CPU Register Tables are critical CPU-related registers considered unlikely to be changed during runtime. They are all write-protected after initialization. This write protection may be temporarily removed if the corresponding CPUy (y=CPU number) Watchdog ENDINIT bit is cleared.

The registers marked “E” in SCU register overview table and other Register Tables are critical system registers considered unlikely to be changed during runtime. They are all write-protected after initialization. This write protection may be temporarily removed if ANY of the CPU Watchdog ENDINIT bits is cleared.

## System Control Units (SCU)

### 9.4.3 Features of the Watchdog Timers

The main features of the WDTs are summarized here.

- 16-bit Watchdog Timer
- Selectable input frequency:  $f_{SPB}/64$ ,  $f_{SPB}/256$  or  $f_{SPB}/16384$
- 16-bit user-definable reload value for normal Watchdog Timer operation, fixed reload value for Time-Out Mode
- Incorporation of the corresponding ENDINIT bit and monitoring of its modifications
- Sophisticated Password Access mechanism with user-definable password fields
- Access Error Detection: Invalid password (during first access) or invalid guard bits (during second access) trigger an alarm request to the SMU
- Temporal and logical monitoring capabilities:
  - Optional Code Sequence checking. An incorrect code sequence signature identification will trigger an alarm request to the SMU
  - Optional Code Execution Time checking. Code execution times out of expected limits will trigger an alarm request to the SMU.
- Overflow Error Detection: An overflow of the WDT counter triggers an alarm request to the SMU
- Watchdog function can be disabled; access protection and ENDINIT bit monitor function remain enabled
- Configurable mechanism to prevent Watchdog reloads after an unserviced safety warning alarm to ensure that unserviced warnings lead to an SMU hardware response

### 9.4.4 The Endinit Functions

There are a number of registers that are usually programmed only once during the initialization sequence of the system or application. Modification of such registers during normal application run can have a severe impact on the overall operation of modules or the entire system.

While the Supervisor Mode and the Access Protection scheme provide a certain level of protection against unintentional modifications, they may not be sufficient to protect against all unintended accesses to system-critical registers.

Additional protection is provided for such registers, called Endinit (“End of initialization”). Endinit is a write-protection scheme that allows writes only during certain times and makes accidental modifications of registers protected by this feature nearly impossible.

The Endinit feature consists of an ENDINIT bit incorporated in each WDT control register. Registers protected via an Endinit determine whether or not writes are enabled. Writes are only enabled if a corresponding ENDINIT = 0 AND Supervisor Mode is active. Write attempts if this condition is not true will be discarded and the register contents will not be modified in this case.

To get the highest level of robustness, writes to the ENDINIT bits are protected by a highly secure access protection scheme implemented in the WDTs. This is a complex procedure, that makes it nearly impossible for ENDINIT bits to be modified unintentionally. In addition, each WDT monitors ENDINIT bit modifications by starting a time-out sequence each time software opens access to the critical registers through clearing the corresponding ENDINIT bit. If the time-out period ends before the corresponding ENDINIT bit is set again, a malfunction of the software is assumed and a Watchdog fault response is generated.

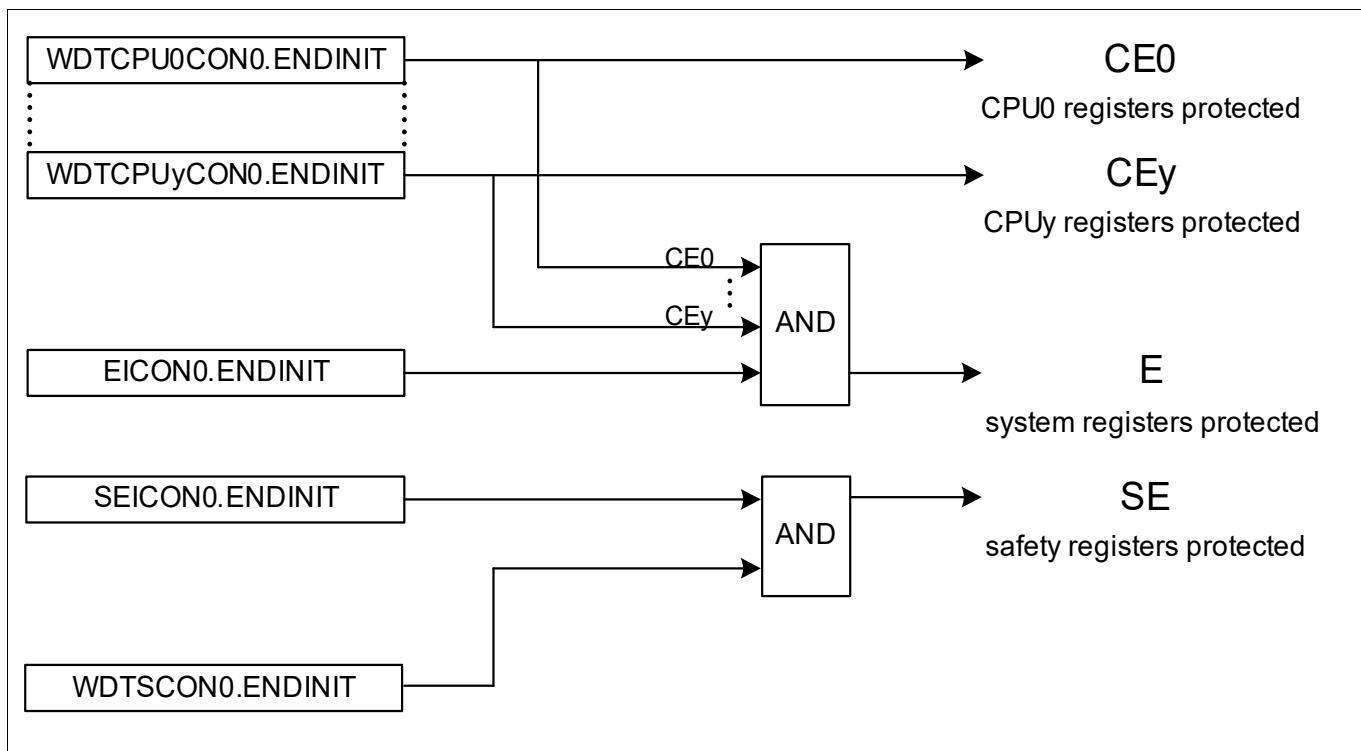
The access-protection scheme and the Endinit time-out operation of the WDTs is described in the following sections. In the register overview tables for each module (including the SCU itself) the registers which are protected via each Endinit type are identified in the column describing write accesses as follows:

- “CEy”- CPU critical registers. Writeable only when CPUy WDT ENDINIT=0 (y=CPU number)

## System Control Units (SCU)

- “E” - System critical registers - Writeable when any (one or more) CPUy Watchdog Timer ENDINIT=0 or EICON0.ENDINIT=0
- “SE” - Safety critical registers - Writeable only when Safety Watchdog Timer ENDINIT=0 or SEICON0.ENDINIT=0
- None of the above - accessible at any time

The options for unlocking the various ENDINIT write-protection modes are shown in [Figure 72](#).



**Figure 72 ENDINIT Control Registers**

Note: *The clearing of an ENDINIT bit takes some time. Accesses to Endinit-protected registers after the clearing of an ENDINIT bit must only be done when the ENDINIT bit is really cleared. As a solution the ENDINIT bit should be read back once before Endinit-protected registers are accessed the first time after bit ENDINIT has been cleared.*

### 9.4.4.1 Password Access to WDTxCON0

A correct password must be written to register WDTxCON0 ( $x = \text{CPUy}$  and  $y = \text{CPU number, or S}$ ) in order to unlock it for modifications. Software must either know the correct password in advance or compute it at runtime. The passwords for each of the Watchdogs Timers ( $x = \text{CPUy}$  and  $y = \text{CPU number, or S}$ ) can be different in order to provide independent watchdog functionality program flows to have independent watchdog functions.

The Safety Watchdog password register WDTSCON0 is protected by the generic SCU protection scheme which allows only configured master(s) to have write access (See [ACCEN10](#)).

CPU-specific Watchdog password registers WDTCPUyCON0 are individually protected such that they may only be written by the corresponding CPUy

A watchdog may be used within a safety application to provide a recovery time period during which software might attempt to recover from a safety alarm warning. To ensure that a CPU fault could not allow a fault to be ignored an option is provided to prevent watchdog unlocking if the Safety Management Unit (SMU) is in the FAULT state. This option may be enabled by bit WDTxCON1.UR .

## System Control Units (SCU)

If the password is valid and the SMU state meets the requirements of the WDTxSR.US bit then WDTxCON0 will be unlocked as soon as the Password Access is completed. The unlocked condition will be indicated by WDTxCON0.LCK = 0. To ensure the correct servicing sequence, a password access is only permitted when the WDTxCON0.LCK bit was set prior to the access.

If an improper password value is written to WDTxCON0 during the Password Access, a Watchdog Access Error condition exists. Bit WDTxSR.AE is set and an alarm request is sent to the Safety Management Unit (SMU).

The 14-bit user-definable password, WDTxCON0.PW, provides additional options for adjusting the password requirements to the application's needs. It can be used, for instance, to detect unexpected software loops, or to monitor the execution sequence of routines.

**Table 265** summarizes the requirements for the password. Various options exist, which are described in more detail below

**Table 265 Password Access Bit Pattern Requirements**

Bit Position	Required Value
[1:0]	Fixed; must be written to 01 <sub>B</sub>
[15:2]	If PAS=0: WDTxCON0.PW[7:2] must be written with inverted current value read from WDTxCON0.PW[7:2] WDTxCON0.PW[15:8] must be written with non-inverted current value read from WDTxCON0.PW[15:8]  If PAS=1: Must be written with Expected Next Sequence Password
[31:16]	If TCS=0: Must be written with current value of user-definable reload value, WDTxCON0.REL If TCS=1: Must be written with inverted estimate of the WDT count value, WDTxSR.TIM. This value must be within +/- WDTxSR.TCT of the actual value

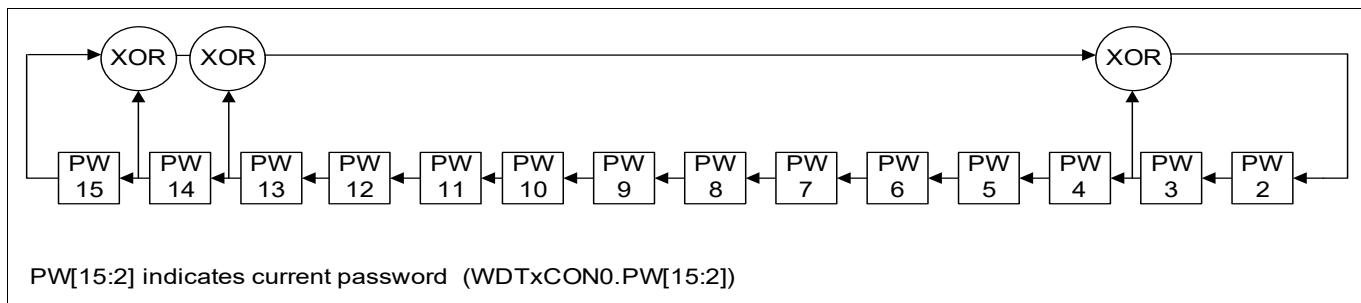
### 9.4.4.1.1 Static Password

In the static password mode (WDTxSR.PAS=0) the password can only be changed by a valid Modify Access. The Password Access has been designed such that it is not possible simply to read the register and re-write it. Some of the password read bits must be inverted (toggled) before being re-written. This prevents a simple malfunction from accidentally unlocking the WDT through a simple read/write sequence.

### 9.4.4.1.2 Automatic Password Sequencing

If automatic password sequencing is enabled (WDTxSR.PAS=1) then the password changes automatically after each password check (i.e. Password Access or Check Access). The Expected Next Password follows a pseudo-random sequence based upon a 14-bit Fibonacci LFSR (Linear Feedback Shift Register) with characteristic polynomial  $x^{14}+x^{13}+x^{12}+x^2+1$ . An initial password (or subsequent manual password updates) can also be provided by Modify Accesses.

## System Control Units (SCU)



**Figure 73 Password Sequencing LFSR**

### 9.4.4.1.3 Time-Independent Password

If time checking is not enabled (WDTxSR.TCS=0) then the REL field of the WDTxCON0 register must be simply re-written with the existing reload value during the Password Access.

### 9.4.4.1.4 Time Check Password

If time checking is enabled (WDTxSR.TCS=1) the REL field of the WDTxCON0 register must be written with an inverted (bit flipped) estimate of the current WDT count value. The acceptable margin of error of this estimate (in WDT clock periods) is specified by the value of WDTxSR.TCT. If the written estimate is outside the range WDTxSR.TIM +/- WDTxSR.TCT, then an SMU alarm condition is indicated. This mechanism can provide a check of the elapsed program execution time since the last WDT restart. Note that a Time Check comparison is still required for a Password or Check Access while the WDT is operating in Time-Out mode (e.g. After accessing ENDINIT-protected registers).

### 9.4.4.2 Check Access to WDTxCON0

A Check Access is identical to a Password Access except that the lock bit is not cleared and a subsequent Modify Access is therefore not allowed. A Check Access does not trigger an SMU Alarm Request provided that the write data requirements are satisfied. A Check Access may only be performed when the LCK bit is set.

This type of access is used for intermediate checkpoints between WDT services. This may be used (e.g. in conjunction with the timestamp count checking feature or sequence passwords) for task sequence or execution time monitoring.

**Table 266 Check Access Bit Pattern Requirements**

Bit Position	Required Value
[1:0]	Fixed; must be written to 11 <sub>B</sub>
[15:2]	If PAS=0: WDTxCON0.PW[7:2] must be written with inverted current value read from WDTxCON0.PW[7:2] WDTxCON0.PW[15:8] must be written with non-inverted current value read from WDTxCON0.PW[15:8]  If PAS=1: Must be written with Expected Next Sequence Password
[31:16]	If TCS=0: Must be written with current value of user-definable reload value, WDTxCON0.REL If TCS=1: Must be written with inverted estimate of the WDT counter WDTxSR.TIM. This value must be within +/- WDTxSR.TCT of the actual value

## System Control Units (SCU)

If an improper value is written to WDTxCON0 (x=CPUy and y=CPU number, or S) during the Check Access, a Watchdog Access Error condition exists. Bit WDTxSR.AE is set and an alarm request is sent to the Safety Management Unit (SMU).

### 9.4.4.3 Modify Access to WDTxCON0

If a WDTxCON0 (x=CPUy and y=CPU number, or S) is successfully unlocked by a Password Access, the following write access to WDTxCON0 can modify it. However, this access must also meet certain requirements in order to be accepted and regarded as valid. **Table 267** lists the required bit patterns. If the access does not follow these rules, a Watchdog Access Error condition is detected, bit WDTxSR.AE is set, and an alarm request is sent to the Safety Management Unit (SMU). After the Modify Access has completed, WDTxCON0.LCK is set again, automatically re-locking WDTxCON0. Before the register can be modified again, a valid Password Access must be executed again.

**Table 267 Modify Access Bit Pattern Requirements**

Bit Position	Value
0	User-definable; desired value for bit WDTxCON0.ENDINIT.
1	Fixed; must be written with $1_B$ .
[15:2]	User-definable; desired value of user-definable password field, WDTxCON0.PW.
[31:16]	User-definable; desired value of user-definable reload value, WDTxCON0.REL.

### 9.4.4.4 Access to Endinit-Protected Registers

If write access to Endinit-protected registers is required during run time, write access can be temporarily re-enabled for a limited time period. Two options are provided:

- Re-enable access to ENDINIT-protected registers with a WDT refresh
- Re-enable access to ENDINIT-protected registers without a WDT refresh

For debugging support the Cerberus module can override all the ENDINIT controls of all WDTs to ease the debug flow. If bit CBS\_OSTATE.ENIDIS is set all ENDINIT protection is disabled independent of the current status configured by the WDTs. If CBS\_OSTATE.ENIDIS is cleared the complete control is within the WDTs.

#### 9.4.4.4.1 Access to Endinit-Protected Registers using WDT

To re-enable access, WDTxCON0 must first be unlocked with a valid Password Access. In the subsequent valid Modify Access, ENDINIT can be cleared. Access to Endinit-protected registers is now open again. However, when WDTxCON0 is unlocked, the WDT is automatically switched to Time-Out Mode. The access window is therefore time-limited. Time-Out Mode is only terminated after ENDINIT has been set again, requiring another Valid Password and Valid Modify Access to WDTxCON0.

#### 9.4.4.4.2 Access to Endinit-Protected Registers without using WDT

In some applications the WDT may not be used and would be disabled (WDTxSR.DS = 1), although this is not recommended.

In other applications, the WDT Timestamp feature may be used and WDT accesses between refreshes would be undesirable.

In these circumstances it is still possible to enable temporary access to Endinit-Protected registers using the ENDINIT Global Control Registers (EICONx).

## System Control Units (SCU)

### 9.4.5 Timer Operation

The timers for Safety Watchdog and CPU0 are automatically active after an Application Reset.

Each 16-bit counter implementing the timer functionality is either triggered with  $f_{SPB} / 64$ ,  $f_{SPB} / 256$  or  $f_{SPB} / 16384$ . The three possible counting rates are controlled via bit WDTxCON1.IRx (x=CPUy and y=CPU number, or S) according to [Table 268](#):

**Table 268 WDT Rate Change via IRx bits (Request) and WDT Rate Indication via SRx bits (Status)**

xR1	xR0	Watchdog Timer Rate (Source Clock divider setting)
0	0	$f_{SPB} / 16384$
0	1	$f_{SPB} / 256$ .
1	0	$f_{SPB} / 64$ .
1	1	Reserved. Do not use.

#### Determining WDT Periods

All WDTs use the SPB clock  $f_{SPB}$ . A clock divider in front of each WDT provides three WDT counter frequencies,  $f_{SPB} / 64$ ,  $f_{SPB} / 256$  and  $f_{SPB} / 16384$ .

The general formula to calculate a Watchdog timeout period is:

$$\text{period} = ((2^{16} - \text{startvalue}) * \text{divider}) / f_{SPB}$$

The parameter startvalue represents the fixed value  $FFFC_H$  for the calculation of the Time-Out Period, and the user-programmable reload value WDTxCON0.REL for the calculation of the Normal Period.

The parameter divider represents the user-programmable source clock division selected by WDTxCON1.IRx, which can be 64, 256 or 16384.

#### 9.4.5.1 Timer Modes

Each Watchdog Timer can operate in one of three different operating modes:

- Time-Out Mode
- Normal Mode
- Disable Mode

The following overview describes these modes and how the WDT changes from one mode to the other.

##### Time-Out Mode

CPU0 WDT Time-Out Mode is entered after an Application Reset. Other CPUs are disabled after Application Reset. Time-Out Mode is also entered when a valid Password Access to register WDTxCON0 is performed. The Time-Out Mode is indicated by bit WDTxSR.TO = 1. The timer is set to  $FFFC_H$  and starts counting upwards. Time-Out Mode can only be exited properly by setting ENDINIT = 1 with a correct access sequence. If an improper access to the WDT is performed, or if the timer overflows before ENDINIT is set, and an alarm request is sent to the Safety Management Unit (SMU).

A proper exit from Time-Out Mode can either be to the Normal or the Disable Mode, depending on the state of the disable request bit WDTxCON1.DR.

## System Control Units (SCU)

### Normal Mode

In Normal Mode ( $DR = 0$ ), the WDT operates in a standard Watchdog fashion. The timer is set to  $WDTxCON0.REL$ , and begins counting up. It has to be serviced before the counter overflows. Servicing is performed through a proper access sequence to the control register  $WDTxCON0$ . This enters the Time-Out Mode.

If the WDT is not serviced before the timer overflows, a system malfunction is assumed. Normal Mode is terminated and an alarm request is sent to the Safety Management Unit (SMU).

### Disabled Mode

All CPU WDTs except CPU0 are in disabled mode after an Application Reset. Disabled Mode is provided for applications which do not require a WDT function. It can be requested from Time-Out Mode when the disable request bit  $WDTxCON1.DR$  is set. The Disabled Mode is entered when was requested AND bit  $WDTxCON0.ENDINIT$  is set. The timer is stopped in this mode. However, disabling the WDT only stops it from performing the standard Watchdog function, eliminating the need for timely service of the WDT. It does not disable Time-Out mode. If an access to register  $WDTxCON0$  is performed in Disabled Mode, Time-Out Mode is entered if the access was valid, and an alarm request is sent to the Safety Management Unit (SMU) if the access was invalid. Thus, the  $ENDINIT$  monitor function as well as (a part of) the system malfunction detection will still be active.

### 9.4.5.2 WDT Alarm Request

SMU alarm requests are generated for three cases:

- Invalid write data during access to register  $WDTxCON0$  ( $x=CPUy$  and  $y=CPU$  number, or S),  $SEICON0$  or  $EICON0$
- Not executing a password access before a timer overflow occurs in the Time-Out Mode of any WDT,  $ENDINIT$  Timeout Counter or Safety  $ENDINIT$  Timeout Counter
- Not serving any WDT before a timer overflow occurs in the Normal Mode

The resulting alarm response (e.g. Trap, Reset etc) is programmable within the Safety Management Unit (SMU).

*Note:* *The WDT itself is reset by any Application Reset.*

### Servicing the Watchdog Timer

If the WDT is used in an application and is enabled ( $WDTxSR.DS = 0$ ), it must be regularly serviced to prevent it from overflowing.

Service is performed in two steps, a valid Password Access followed by a valid Modify Access. The valid Password Access to  $WDTxCON0$  automatically switches the WDT to Time-Out Mode. Thus, the Modify Access must be performed before the time-out expires or an alarm request will be sent to the Safety Management Unit (SMU).

During the next Modify Access, the strict requirement is that  $WDTxCON0.ENDINIT$  is written with 1.

*Note:*  *$ENDINIT$  must be written with 1 to perform a proper service, even if it is already set to 1.*

Changes to the reload value  $WDTxCON0.REL$ , or the user-definable password  $WDTxCON0.PW$ , are not required. When a WDT service is properly executed, Time-Out Mode is terminated, and the WDT switches back to its former mode of operation, and the WDT service is complete.

Check Accesses are optional and may be performed at any time when the WDT is locked.

### 9.4.5.3 WDT Operation During Power-Saving Modes

If one or more of the CPU are in Idle Mode, they cannot service a WDT because no software is running. Excluding the case where the system is running normally, a strategy for managing WDTs is needed while a CPU is in Idle

## System Control Units (SCU)

Mode. There are two ways to manage a WDT in these cases. Firstly, the Watchdog can be disabled before idling the CPU. The disadvantage of this is that the system will no longer be monitored during the idle period.

A better approach to this problem relies upon the wake-up feature of the WDTs. Whenever CPUy is put in Idle or Sleep Mode and the WDT is not disabled, the CPUy is woken at regular intervals. When WDTx (x=CPUy and y= CPU number) changes its count value (WDTxSR.TIM) from  $7FFF_H$  to  $8000_H$ , CPUy is woken and continues execution at the instruction following the instruction that was executed before entering the Idle or Sleep Mode.

**Note:** *Before switching into a non-running power-management mode, software should perform a Watchdog service sequence. At the Modify Access, the Watchdog reload value, WDTxCON0.REL, should be programmed such that the wake-up occurs after a period which best meets application requirements. The maximum period between two CPU wake-ups is one-half of the maximum WDT period.*

### 9.4.5.4 Suspend Mode Support

During a debug session the Watchdog or SEI/EI Timeout Counter functionality might lead to unintended alarms.

By default, the WDTs and SEI/EI Timeout Counters are disabled when OCDS is enabled (and also during start-up). However, it is possible to enable WDTs and SEI/EI Timeout Counter even when OCDS is enabled if CBS\_OSTATE.WDTSUS=1.

CPUy WDTs may only be re-enabled if the corresponding CPUySUSOUT is inactive. In WDTS, SEICON and EICON CBS\_TLS.TL1 is used for this purpose.

For safety reasons it is essential that WDT ignores CPUySUSOUT when OCDS is disabled. WDTS, EICON/SEICON shall ignore CBS\_TLS.TL1 when OCDS is disabled.

**Table 269 CPUy WDT Suspend State**

STCON.STP	WDTxSR.DS (x=CPUy)	CBS_OSTATE.OEN	CBS_OSTATE.WDTSUS	CPUy SUSOUT	WDTCPUy State
0	X	X	X	X	Stopped
1	1	X	X	X	Stopped
1	0	0	X	X	Running
1	0	1	0	X	Stopped
1	0	1	1	0	Running
1	0	1	1	1	Stopped

**Table 270 Safety WDT, SEICON and EICON Endinit Counter Suspend State**

STCON.STP	WDTxSR.DS	CBS_OSTATE.OEN	CBS_OSTATE.WDTSUS	CBS_TLS. TL1	WDTS, SEICON, EICON State
0	X	X	X	X	Stopped
1	1	X	X	X	Stopped
1	0	0	X	X	Running
1	0	1	0	X	Stopped

---

**System Control Units (SCU)**
**Table 270 Safety WDT, SEICON and EICON Endinit Counter Suspend State (cont'd)**

<b>STCON.STP</b>	<b>WDTxSR.DS</b>	<b>CBS_OSTATE.OEN</b>	<b>CBS_OSTATE.WDTSUS</b>	<b>CBS_TLS. TL1</b>	<b>WDTS, SEICON, EICON State</b>
1	0	1	1	0	Running
1	0	1	1	1	Stopped

## System Control Units (SCU)

### 9.4.6 Watchdog Timer Registers

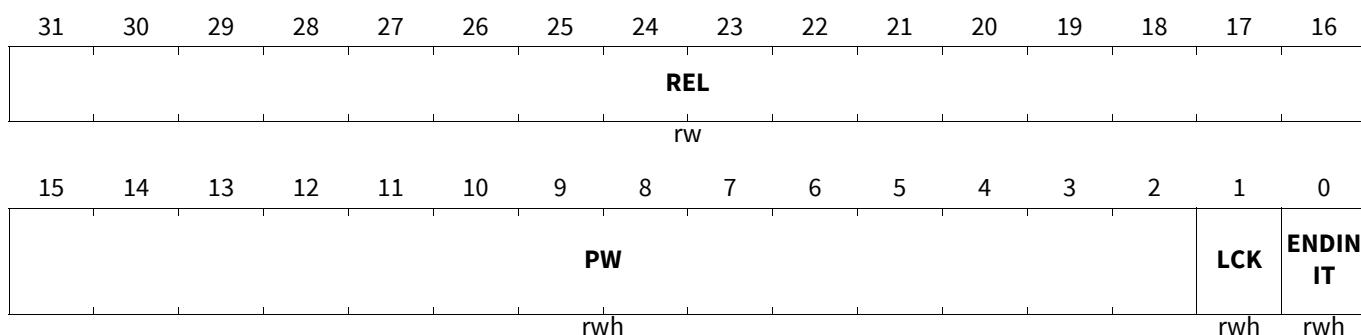
#### Safety WDT Control Register 0

This register is written with check data in Check Accesses and Password Accesses. It also stores the timer reload value, password update and the corresponding End-of-Initialization (ENDINIT) control bit during a Modify Access.

There is a WDT<sub>x</sub>CON0 register for each Watchdog x (x=CPUy with y= CPU number and x=S). For the Safety Watchdog Timer is x=S.

#### WDT<sub>S</sub>CON0

**Safety WDT Control Register 0** **(02A8<sub>H</sub>)** **Application Reset Value: FFFC 000E<sub>H</sub>**



Field	Bits	Type	Description
<b>ENDINIT</b>	0	rwh	<p><b>End-of-Initialization Control Bit</b></p> <p>This bit must be written with a ‘1’ during a Password Access or Check Access (although this write is only used for the password-protection mechanism and is not stored). This bit must be written with the required ENDINIT update value during a Modify Access.</p> <p>This bit may be used to access registers with “SE” protection, but the alternate register SEICON0.ENDINIT is recommended for this purpose so that the Watchdog Timer is not affected.</p> <p>0<sub>B</sub> Access to Endinit-protected registers is permitted (default after ApplicationReset)</p> <p>1<sub>B</sub> Access to Endinit-protected registers is not permitted.</p>
<b>LCK</b>	1	rwh	<p><b>Lock Bit to Control Access to WDT<sub>x</sub>CON0</b></p> <p>The current value of LCK is controlled by hardware. It is cleared after a valid Password Access to WDT<sub>x</sub>CON0 when WDT<sub>x</sub>SR.US is 0 (or when WDT<sub>x</sub>SR.US is 1 and the SMU is in RUN mode), and it is automatically set again after a valid Modify Access to WDT<sub>x</sub>CON0. During a write to WDT<sub>x</sub>CON0, the value written to this bit is only used for the password-protection mechanism and is not stored.</p> <p>This bit must be cleared during a Password Access to WDT<sub>x</sub>CON0, and set during a Modify Access to WDT<sub>x</sub>CON0.</p> <p>A Check Access does not clear LCK.</p> <p>0<sub>B</sub> Register WDT<sub>x</sub>CON0 is unlocked</p> <p>1<sub>B</sub> Register WDT<sub>x</sub>CON0 is locked (default after ApplicationReset)</p>

## System Control Units (SCU)

Field	Bits	Type	Description
PW	15:2	rwh	<p><b>User-Definable Password Field for Access to WDTxCON0</b></p> <p>This bit field is written with an initial password value during a Modify Access.</p> <p>A read from this bitfield returns this initial password, but bits [7:2] are inverted (toggled) to ensure that a simple read/write is not sufficient to service the WDTx.</p> <p>If corresponding WDTxSR.PAS = 0 then this bit field must be written with its current contents during a Password Access or Check Access.</p> <p>If corresponding WDTxSR.PAS = 1 then this bit field must be written with the next password in the LFSR sequence during a Password Access or Check Access</p> <p>The default password after Application Reset is 00000000111100<sub>B</sub></p>
REL	31:16	rw	<p><b>Reload Value for the WDT (also Time Check Value)</b></p> <p>The reload value can be changed during a Modify Access to WDTxCON0 (Default after ApplicationReset is FFFC<sub>H</sub>). If the Watchdog Timer is enabled and in Normal Timer Mode, it will start counting from this value after a correct Watchdog service.</p> <p>A read from this bitfield always returns the current reload value.</p> <p>During a Password Access or a Check Access this bitfield may be used for additional checks. Writes during such checks have no effect upon the reload value.</p> <p>If corresponding WDTxSR.TCS=0 then this bit field must be written with its current contents during a Password Access or Check Access.</p> <p>If corresponding WDTxSR.TCS=1 then this bit field must be written with an inverted estimate of the current WDTxSR.TIM value during a Password Access or Check Access.</p>

### CPUy WDT Control Register 0

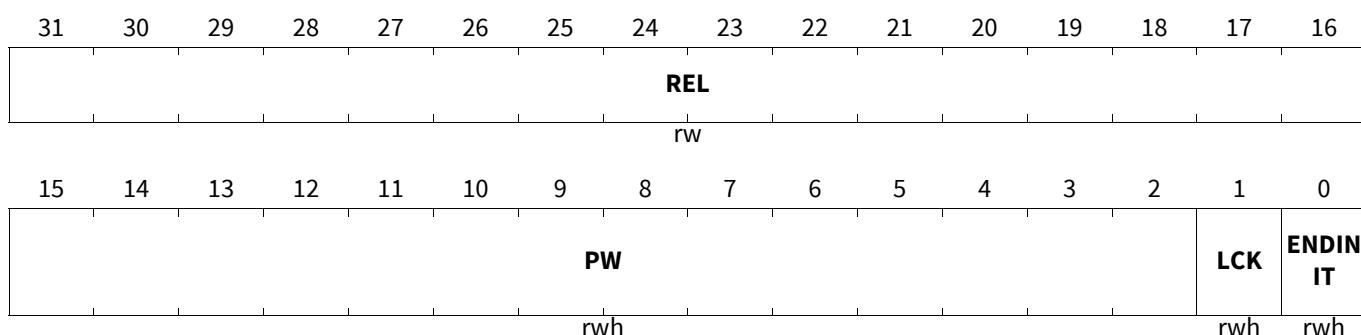
This register is written with check data in Check Accesses and Password Accesses. It also stores the timer reload value, password update and the corresponding End-of-Initialization (ENDINIT) control bit during a Modify Access. There is a WDTxCON0 register for each Watchdog x (x=CPUy with y= CPU number and x=S). For the CPU Watchdog Timers is x=CPUy with y=CPU number.

#### WDTCPUyCON0 (y=0-5)

#### CPUy WDT Control Register 0

(024C<sub>H</sub>+y\*12)

Reset Value: [Table 271](#)



**System Control Units (SCU)**

Field	Bits	Type	Description
<b>ENDINIT</b>	0	rwh	<p><b>End-of-Initialization Control Bit</b></p> <p>This bit must be written with a ‘1’ during a Password Access or Check Access (although this write is only used for the password-protection mechanism and is not stored). This bit must be written with the required ENDINIT update value during a Modify Access.</p> <p>This bit is intended for accessing registers with “CEy” protection (y=CPU number). It may also be used to access registers with “E” protection, but the alternate register EICON0.ENDINIT is recommended for this purpose so that the Watchdog Timer is not affected.</p> <p><math>0_B</math> Access to Endinit-protected registers is permitted (default after ApplicationReset)  <math>1_B</math> Access to Endinit-protected registers is not permitted.</p>
<b>LCK</b>	1	rwh	<p><b>Lock Bit to Control Access to WDTxCON0</b></p> <p>The current value of LCK is controlled by hardware. It is cleared after a valid Password Access to WDTxCON0 when WDTxSR.US is 0 (or when WDTxSR.US is 1 and the SMU is in RUN mode), and it is automatically set again after a valid Modify Access to WDTxCON0. During a write to WDTxCON0, the value written to this bit is only used for the password-protection mechanism and is not stored.</p> <p>This bit must be cleared during a Password Access to WDTxCON0, and set during a Modify Access to WDTxCON0.</p> <p>A Check Access does not clear LCK.</p> <p><math>0_B</math> Register WDTxCON0 is unlocked  <math>1_B</math> Register WDTxCON0 is locked (default after ApplicationReset)</p>
<b>PW</b>	15:2	rwh	<p><b>User-Definable Password Field for Access to WDTxCON0</b></p> <p>This bit field is written with an initial password value during a Modify Access.</p> <p>A read from this bitfield returns this initial password, but bits [7:2] are inverted (toggled) to ensure that a simple read/write is not sufficient to service the WDTx.</p> <p>If corresponding WDTxSR.PAS = 0 then this bit field must be written with its current contents during a Password Access or Check Access.</p> <p>If corresponding WDTxSR.PAS = 1 then this bit field must be written with the next password in the LFSR sequence during a Password Access or Check Access</p> <p>The default password after Application Reset is 00000000111100<sub>B</sub></p>

## System Control Units (SCU)

Field	Bits	Type	Description
<b>REL</b>	31:16	rw	<p><b>Reload Value for the WDT (also Time Check Value)</b></p> <p>The reload value can be changed during a Modify Access to WDTxCON0 (Default after ApplicationReset is <math>\text{FFFC}_H</math>). If the Watchdog Timer is enabled and in Normal Timer Mode, it will start counting from this value after a correct Watchdog service.</p> <p>A read from this bitfield always returns the current reload value.</p> <p>During a Password Access or a Check Access this bitfield may be used for additional checks. Writes during such checks have no effect upon the reload value.</p> <p>If corresponding WDTxSR.TCS=0 then this bit field must be written with its current contents during a Password Access or Check Access.</p> <p>If corresponding WDTxSR.TCS=1 then this bit field must be written with an inverted estimate of the current WDTxSR.TIM value during a Password Access or Check Access.</p>

**Table 271 Reset Values of WDTCPUyCON0 (y=0-5)**

Reset Type	Reset Value	Note
Application Reset	$\text{FFFC } 000E_H$	WDTCPU0CON0
Application Reset	$\text{FFFC } 000F_H$	WDTCPUyCON0 (y=1-5)

### Safety WDT Control Register 1

There is a WDTxCON1 register for each Watchdog Timer x (x=CPUy with y=CPU number and x=S). For the Safety Watchdog Timer is x=S. The register WDTxCON1 manages operation of the Safety WDT. It includes the disable request, password configuration and frequency selection bits. Each WDTxCON1 register is write-protected by the Safety ENDINIT (SE).

WDTSCON1 has an additional bit CLRIRF which can be used to clear the internal reset status used to detect double SMU (e.g. WDT) resets.

#### WDTSCON1

#### Safety WDT Control Register 1 (02AC<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
TCTR								TCR	PAR	UR	IR1	0	DR	IR0	0	CLRIRF

**System Control Units (SCU)**

Field	Bits	Type	Description
<b>CLRIRF</b>	0	rwh	<p><b>Clear Internal Reset Flag</b></p> <p>This bit is used to request a clear of the internal flag which indicates whether a previous SMU reset has already been requested. After modification, the internal flag is only cleared when Safety Endinit (SE) is re-asserted. As long as Safety ENDINIT(SE) is not asserted, the internal flag is unchanged and continues to determine the response to a further SMU reset request. When Safety ENDINIT is reasserted, the internal flag is cleared together with this bit.</p> <p>0<sub>B</sub> No action 1<sub>B</sub> Request to clear the internal previous-SMU-reset flag</p>
<b>IR0</b>	2	rw	<p><b>Input Frequency Request Control - IR1,IR0</b></p> <p>Bit IR0 and IR1 should be programmed together to determine the WDTx timer frequency.</p> <p>WDTxSR.IS0 and WDTxSR.IS1 are updated by these bits only when Safety ENDINIT (SE) is re-asserted. As long as Safety ENDINIT is de-asserted, WDTxSR.IS0 and WDTxSR.IS1 control the current input frequency of the Safety Watchdog Timer. When Safety ENDINIT is reasserted, WDTxSR.IS0 and WDTxSR.IS1 are updated with the values of IR0 and IR1.</p> <p>0<sub>B</sub> If Bit IR1=0 Request to set input frequency to <math>f_{SPB}/16384</math>. Elseif Bit IR1=1 Request to set input frequency to <math>f_{SPB}/64</math>. 1<sub>B</sub> If Bit IR1=0 Request to set input frequency to <math>f_{SPB}/256</math>. Elseif Bit IR1=1 Reserved. Do not use.</p>
<b>DR</b>	3	rw	<p><b>Disable Request Control Bit</b></p> <p>This bit can only be modified when Safety ENDINIT (SE) is de-asserted. WDTxSR.DS is updated when Safety ENDINIT is re-asserted. As long as Safety ENDINIT is de-asserted, bit WDTxSR.DS controls the current enable/disable status of the WDTx. When Safety ENDINIT is reasserted, WDTxSR.DS is updated with the state of DR.</p> <p>0<sub>B</sub> Request to enable the WDTx 1<sub>B</sub> Request to disable the WDTx</p>
<b>IR1</b>	5	rw	<p><b>Input Frequency Request Control</b></p> <p>Bit IR0 and IR1 should be programmed together to determine the WDTx timer frequency</p> <p>WDTxSR.IS0 and WDTxSR.IS1 are updated by these bits only when Safety ENDINIT (SE) is re-asserted. As long as Safety ENDINIT is de-asserted, WDTxSR.IS0 and WDTxSR.IS1 control the current input frequency of the Safety Watchdog Timer. When Safety ENDINIT is reasserted, WDTxSR.IS0 and WDTxSR.IS1 are updated with the values of IR0 and IR1.</p> <p>0<sub>B</sub> If Bit IR0=0 Request to set input frequency to <math>f_{SPB}/16384</math>. Elseif Bit IR0=1 Request to set input frequency to <math>f_{SPB}/256</math>. 1<sub>B</sub> If Bit IR0=0 Request to set input frequency to <math>f_{SPB}/64</math>. Elseif Bit IR0=1 Reserved. Do not use.</p>

## System Control Units (SCU)

Field	Bits	Type	Description
UR	6	rw	<p><b>Unlock Restriction Request Control Bit</b></p> <p>This bit can only be modified when Safety ENDINIT (SE) is de-asserted. WDTxSR.US is updated when Safety ENDINIT is reasserted. As long as the Safety ENDINIT is cleared, bit WDTxSR.US controls whether unlocking is possible at all times or only when the SMU is not in the FAULT state.</p> <p>When Safety ENDINIT is reasserted, WDTxSR.US is updated with the state of UR.</p> <p>0<sub>B</sub> Request to disable SMU restriction of WDTx unlock 1<sub>B</sub> Request to enable SMU restriction of WDTx unlock</p>
PAR	7	rw	<p><b>Password Auto-sequence Request Bit</b></p> <p>This bit can only be modified when Safety ENDINIT is de-asserted. WDTxSR.PAS is updated when Safety ENDINIT is reasserted. As long as Safety ENDINIT is de-asserted, bit WDTxSR.PAS controls password sequencing. When Safety ENDINIT is reasserted, WDTxSR.PAS is updated with the state of PAR.</p> <p>0<sub>B</sub> Request no automatic change of password after each Modify Access or Check Access 1<sub>B</sub> Request automatic sequence of password after each Modify Access or Check Access</p>
TCR	8	rw	<p><b>Counter Check Request Bit</b></p> <p>This bit can only be modified when Safety ENDINIT (SE) is de-asserted. WDTxSR.TCS is updated when Safety ENDINIT is re-asserted. As long as Safety ENDINIT is de-asserted, bit WDTxSR.TCS controls whether counter check is enabled. When Safety ENDINIT is reasserted, WDTxSR.TCS is updated with the state of TCR</p> <p>0<sub>B</sub> Request to check only that REL field is written with existing REL value during Modify Access or Check Access 1<sub>B</sub> Request to check that REL field is written with correct TIM Count (within tolerance of WDTxSR.TCT) during Modify Access or Check Access</p>
TCTR	15:9	rw	<p><b>Timer Check Tolerance Request</b></p> <p>This bit can only be modified when Safety ENDINIT is de-asserted. WDTxSR.TCT is updated when Safety ENDINIT is reasserted. As long as Safety ENDINIT is de-asserted, bit WDTxSR.TCT controls the tolerance of timer checks. When Safety ENDINIT is re-asserted, WDTxSR.TCT is updated with the state of TCTR.</p>
0	1, 4, 31:16	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

## CPUy WDT Control Register 1

The register WDTxCON1 manages the operation of CPU Watchdog Timer WDTx with x=CPUy and y=CPU number. It includes the disable request, password configuration and frequency selection bits. Each WDTxCON1 register is protected by the corresponding ENDINIT which it controls.

## System Control Units (SCU)

## WDTCPuCON1 (y=0-5)

## CPUy WDT Control Register 1

(0250<sub>H</sub>+y\*12)Reset Value: [Table 272](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
TCTR								TCR	PAR	UR	IR1	0	DR	IR0	0	0
rw				rw		rw		rw		rw		rw		r		

Field	Bits	Type	Description
IR0	2	rw	<p><b>Input Frequency Request Control - IR1,IR0</b></p> <p>Bit IR0 and IR1 should be programmed together to determine the WDT timer frequency.</p> <p>These bits can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.IS0 and WDTxSR.IS1 are updated by these bit only when ENDINIT is set again. As long as ENDINIT is cleared, WDTxSR.IS0 and WDTxSR.IS1 controls the current input frequency of the Watchdog Timer. When ENDINIT is set again, WDTxSR.IS0 and WDTxSR.IS1 are updated with the values of IR0 and IR1.</p> <p>0<sub>B</sub> If Bit IR1=0 Request to set input frequency to f<sub>SPB</sub>/16384. Elseif Bit IR1=1 Request to set input frequency to f<sub>SPB</sub>/64.</p> <p>1<sub>B</sub> If Bit IR1=0 Request to set input frequency to f<sub>SPB</sub>/256. Elseif Bit IR1=1 Reserved. Do not use</p>
DR	3	rw	<p><b>Disable Request Control Bit</b></p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.DS is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTxSR.DS controls the current enable/disable status of the WDTx. When the ENDINIT is set again with a Valid Modify Access, WDTxSR.DS is updated with the state of DR.</p> <p>0<sub>B</sub> Request to enable the WDTx</p> <p>1<sub>B</sub> Request to disable the WDTx</p>
IR1	5	rw	<p><b>Input Frequency Request Control</b></p> <p>Bit IR0 and IR1 should be programmed together to determine the WDTx timer frequency</p> <p>These bits can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.IS0 and WDTxSR.IS1 are updated by these bit only when ENDINIT is set again. As long as ENDINIT is cleared, WDTxSR.IS0 and WDTxSR.IS1 control the current input frequency of the Watchdog Timer. When ENDINIT is set again, WDTxSR.IS0 and WDTxSR.IS1 are updated with the values of IR0 and IR1.</p> <p>0<sub>B</sub> If Bit IR0=0 Request to set input frequency to f<sub>SPB</sub>/16384. Elseif Bit IR0=1 Request to set input frequency to f<sub>SPB</sub>/256.</p> <p>1<sub>B</sub> If Bit IR0=0 Request to set input frequency to f<sub>SPB</sub>/64. Elseif Bit IR0=1 Reserved. Do not use.</p>

## System Control Units (SCU)

Field	Bits	Type	Description
<b>UR</b>	6	rw	<p><b>Unlock Restriction Request Control Bit</b></p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.US is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTxSR.US controls whether unlocking is possible at all times or only when the SMU is not in the FAULT state. When the ENDINIT is set again with a Valid Modify Access, WDTxSR.US is updated with the state of UR.</p> <p>0<sub>B</sub> Request to disable SMU restriction of WDTx unlock 1<sub>B</sub> Request to enable SMU restriction of WDTx unlock</p>
<b>PAR</b>	7	rw	<p><b>Password Auto-sequence Request Bit</b></p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.PAS is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTxSR.PAS controls password sequencing. When the ENDINIT is set again with a Valid Modify Access, WDTxSR.PAS is updated with the state of PAR.</p> <p>0<sub>B</sub> Request no automatic change of password after each Modify Access or Check Access 1<sub>B</sub> Request automatic sequence of password after each Modify Access or Check Access</p>
<b>TCR</b>	8	rw	<p><b>Counter Check Request Bit</b></p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.TCS is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTxSR.TCS controls whether counter check is enabled. When the ENDINIT is set again with a Valid Modify Access, WDTxSR.TCS is updated with the state of TCR</p> <p>0<sub>B</sub> Request to check only that REL field is written with existing REL value during Modify Access or Check Access 1<sub>B</sub> Request to check that REL field is written with correct TIM Count (within tolerance of WDTxSR.TCT) during Modify Access or Check Access</p>
<b>TCTR</b>	15:9	rw	<p><b>Timer Check Tolerance Request</b></p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.TCT is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTxSR.TCT controls the tolerance of timer checks. When the ENDINIT is set again with a Valid Modify Access, WDTxSR.TCT is updated with the state of TCTR</p>
<b>0</b>	0, 1, 4, 31:16	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 272 Reset Values of WDTCPUyCON1 (y=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	WDTCPU0CON1
Application Reset	0000 0008 <sub>H</sub>	WDTCPUyCON1 (y=1-5)

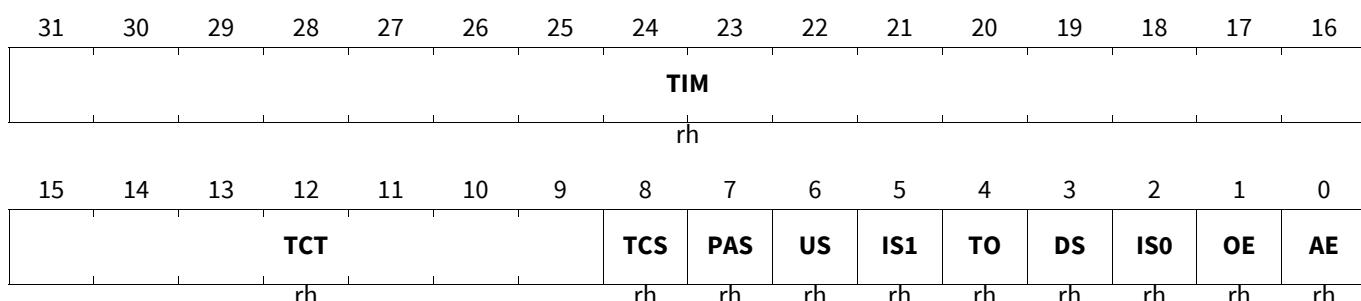
## System Control Units (SCU)

### Safety WDT Status Register

The WDTxSR registers show the current state of each WDTx. For the Safety Watchdog Timer is x=S. Status include bits indicating Time-Out, enable/disable status, input clock status, and access error status.

#### WDTSSR

**Safety WDT Status Register (02B0<sub>H</sub>) Application Reset Value: FFFC 0010<sub>H</sub>**



Field	Bits	Type	Description
AE	0	rh	<b>Watchdog Access Error Status Flag</b> This bit is set when an illegal Password Access or Modify Access to register WDTxCON0 was attempted. This bit is only cleared when WDTxCON0.ENDINIT is set during a valid Modify Access 0 <sub>B</sub> No Watchdog access error 1 <sub>B</sub> A Watchdog access error has occurred
OE	1	rh	<b>Watchdog Overflow Error Status Flag</b> This bit is set when the WDTx overflows from FFFF <sub>H</sub> to 0000 <sub>H</sub> . This bit is only cleared when WDTxCON0.ENDINIT is set to 1 during a valid Modify Access. 0 <sub>B</sub> No Watchdog overflow error 1 <sub>B</sub> A Watchdog overflow error has occurred
IS0	2	rh	<b>Watchdog Input Clock Status - IS1,IS0</b> Bit IS0 and IS1 should be programmed together. These bits indicate the current WDTx clock rate. These bits are updated with the state of bits WDTxCON1.IR0 and WDTxCON1.IR1 after WDTxCON0.ENDINIT is written with 1 during a valid Modify Access to register WDTxCON0. 0 <sub>B</sub> If Bit IS1=0 WDTx counter frequency is f <sub>SPB</sub> /16384. Elseif Bit IS1=1 WDTx counter frequency is f <sub>SPB</sub> /64. 1 <sub>B</sub> If Bit IS1=0 Request to set input frequency to f <sub>SPB</sub> /256. Elseif Bit IS1=1 Reserved. Do not use.
DS	3	rh	<b>Watchdog Enable/Disable Status Flag</b> This bit is updated with the state of bit WDTxCON1.DR (after WDTxCON0.ENDINIT is set during a Valid Modify Access to register WDTxCON0) and it is cleared when Time-Out mode is entered. 0 <sub>B</sub> WDTx is enabled (default after ApplicationReset) 1 <sub>B</sub> WDTx is disabled

**System Control Units (SCU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>TO</b>	4	rh	<p><b>Watchdog Time-Out Mode Flag</b></p> <p>This bit is set when Time-Out Mode is entered. It is automatically cleared when Time-Out Mode is left.</p> <p>0<sub>B</sub> The Watchdog is not operating in Time-Out Mode 1<sub>B</sub> The Watchdog is operating in Time-Out Mode (default after ApplicationReset)</p>
<b>IS1</b>	5	rh	<p><b>Watchdog Input Clock Status</b></p> <p>Bit ISO and IS1 should be programmed together. These bits indicate the current WDTx clock rate. These bits are updated with the state of bits WDTxCON1.IR0 and WDTxCON1.IR1 after WDTxCON0.ENDINIT is written with 1 during a valid Modify Access to register WDTxCON0.</p> <p>0<sub>B</sub> If Bit IS0=0 WDTx counter frequency is f<sub>SPB</sub>/16384. Elseif Bit IS0=1 WDTx counter frequency is f<sub>SPB</sub>/256. 1<sub>B</sub> If Bit IS0=0 WDTx counter frequency is f<sub>SPB</sub>/64. Elseif Bit IS0=1 Reserved. Do not use.</p>
<b>US</b>	6	rh	<p><b>SMU Unlock Restriction Status Flag</b></p> <p>WDTxCON0.LCK will not be unlocked by a valid Password Access if this bit is ‘1’ and the SMU is not in the FAULT state</p> <p>0<sub>B</sub> WDTx unlock permitted at any time 1<sub>B</sub> WDTx unlock only permitted when the SMU is in not the FAULT state.</p>
<b>PAS</b>	7	rh	<p><b>Password Auto-sequence Status Flag</b></p> <p>This bit is updated with the state of bit WDTxCON1.PAR after WDTxCON0.ENDINIT is written with 1 during a valid Modify Access to register WDTxCON0.</p> <p>0<sub>B</sub> No change of password after each Modify Access or Check Access 1<sub>B</sub> Automatically sequence the password after each Modify Access or Check Access</p>
<b>TCS</b>	8	rh	<p><b>Timer Check Status Flag</b></p> <p>This bit is updated with the state of bit WDTxCON1.TCR after WDTxCON0.ENDINIT is written with 1 during a Valid Modify Access to register WDTxCON0.</p> <p>0<sub>B</sub> Check only that REL field is written with existing REL value during Modify Access or Check Access 1<sub>B</sub> Check that REL field is written with inverted estimated TIM value (within +/- TCT value) during Password Access or Check Access</p>
<b>TCT</b>	15:9	rh	<p><b>Timer Check Tolerance</b></p> <p>This field determines the tolerance of the timer check during Password or Check Access (See TCS).This bit is updated with the state of bit WDTxCON1.TCTR after WDTxCON0.ENDINIT is written with 1 during a Valid Modify Access to register WDTxCON0.</p>
<b>TIM</b>	31:16	rh	<p><b>Timer Value</b></p> <p>Reflects the current content of the WDTx.</p>

## System Control Units (SCU)

### CPUy WDT Status Register

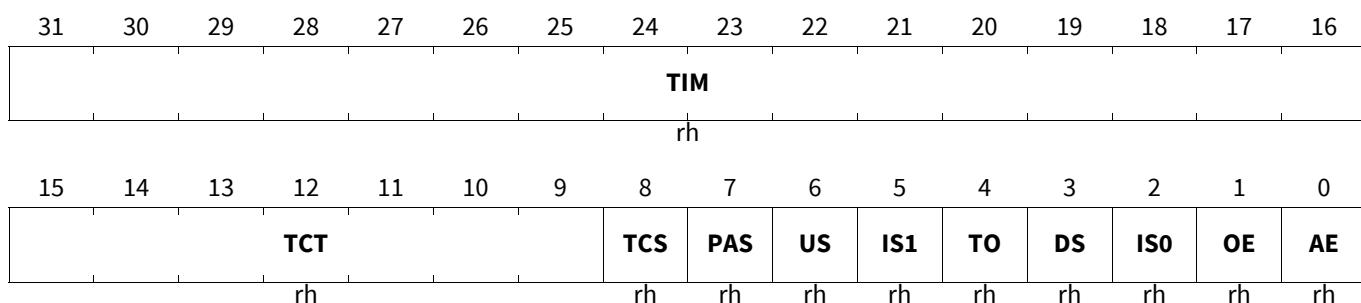
The WDTxSR registers show the current state of each WDTx. For the CPU Watchdog Timer is x=CPUy with y=CPU number. Status include bits indicating Time-Out, enable/disable status, input clock status, and access error status.

#### WDTCPUsR (y=0-5)

#### CPUy WDT Status Register

(0254<sub>H</sub>+y\*12)

Reset Value: [Table 273](#)



Field	Bits	Type	Description
<b>AE</b>	0	rh	<b>Watchdog Access Error Status Flag</b> This bit is set when an illegal Password Access or Modify Access to register WDTxCON0 was attempted. This bit is only cleared when WDTxCON0.ENDINIT is set during a valid Modify Access 0 <sub>B</sub> No Watchdog access error 1 <sub>B</sub> A Watchdog access error has occurred
<b>OE</b>	1	rh	<b>Watchdog Overflow Error Status Flag</b> This bit is set when the WDTx overflows from FFFF <sub>H</sub> to 0000 <sub>H</sub> . This bit is only cleared when WDTxCON0.ENDINIT is set to 1 during a valid Modify Access. 0 <sub>B</sub> No Watchdog overflow error 1 <sub>B</sub> A Watchdog overflow error has occurred
<b>ISO</b>	2	rh	<b>Watchdog Input Clock Status - IS1,ISO</b> Bit ISO and IS1 should be programmed together. These bits indicate the current WDTx clock rate. These bits are updated with the state of bits WDTxCON1.IR0 and WDTxCON1.IR1 after WDTxCON0.ENDINIT is written with 1 during a valid Modify Access to register WDTxCON0. 0 <sub>B</sub> If Bit IS1=0 WDTx counter frequency is f <sub>SPB</sub> /16384. Elseif Bit IS1=1 WDTx counter frequency is f <sub>SPB</sub> /64. 1 <sub>B</sub> If Bit IS1=0 Request to set input frequency to f <sub>SPB</sub> /256. Elseif Bit IS1=1 Reserved. Do not use.
<b>DS</b>	3	rh	<b>Watchdog Enable/Disable Status Flag</b> This bit is updated with the state of bit WDTxCON1.DR (after WDTxCON0.ENDINIT is set during a Valid Modify Access to register WDTxCON0) and it is cleared when Time-Out mode is entered. 0 <sub>B</sub> WDTx is enabled (default after ApplicationReset) 1 <sub>B</sub> WDTx is disabled

**System Control Units (SCU)**

Field	Bits	Type	Description
<b>TO</b>	4	rh	<p><b>Watchdog Time-Out Mode Flag</b></p> <p>This bit is set when Time-Out Mode is entered. It is automatically cleared when Time-Out Mode is left.</p> <p>0<sub>B</sub> The Watchdog is not operating in Time-Out Mode 1<sub>B</sub> The Watchdog is operating in Time-Out Mode (default after ApplicationReset)</p>
<b>IS1</b>	5	rh	<p><b>Watchdog Input Clock Status</b></p> <p>Bit ISO and IS1 should be programmed together. These bits indicate the current WDTx clock rate. These bits are updated with the state of bits WDTxCON1.IR0 and WDTxCON1.IR1 after WDTxCON0.ENDINIT is written with 1 during a valid Modify Access to register WDTxCON0.</p> <p>0<sub>B</sub> If Bit IS0=0 WDTx counter frequency is f<sub>SPB</sub>/16384. Elseif Bit IS0=1 WDTx counter frequency is f<sub>SPB</sub>/256. 1<sub>B</sub> If Bit IS0=0 WDTx counter frequency is f<sub>SPB</sub>/64. Elseif Bit IS0=1 Reserved. Do not use.</p>
<b>US</b>	6	rh	<p><b>SMU Unlock Restriction Status Flag</b></p> <p>WDTxCON0.LCK will not be unlocked by a valid Password Access if this bit is ‘1’ and the SMU is not in the FAULT state</p> <p>0<sub>B</sub> WDTx unlock permitted at any time 1<sub>B</sub> WDTx unlock only permitted when the SMU is not in the FAULT state.</p>
<b>PAS</b>	7	rh	<p><b>Password Auto-sequence Status Flag</b></p> <p>This bit is updated with the state of bit WDTxCON1.PAR after WDTxCON0.ENDINIT is written with 1 during a valid Modify Access to register WDTxCON0.</p> <p>0<sub>B</sub> No change of password after each Modify Access or Check Access 1<sub>B</sub> Automatically sequence the password after each Modify Access or Check Access</p>
<b>TCS</b>	8	rh	<p><b>Timer Check Status Flag</b></p> <p>This bit is updated with the state of bit WDTxCON1.TCR after WDTxCON0.ENDINIT is written with 1 during a Valid Modify Access to register WDTxCON0.</p> <p>0<sub>B</sub> Check only that REL field is written with existing REL value during Modify Access or Check Access 1<sub>B</sub> Check that REL field is written with inverted estimated TIM value (within +/- TCT value) during Password Access or Check Access</p>
<b>TCT</b>	15:9	rh	<p><b>Timer Check Tolerance</b></p> <p>This field determines the tolerance of the timer check during Password or Check Access (See TCS).This bit is updated with the state of bit WDTxCON1.TCTR after WDTxCON0.ENDINIT is written with 1 during a Valid Modify Access to register WDTxCON0.</p>
<b>TIM</b>	31:16	rh	<p><b>Timer Value</b></p> <p>Reflects the current content of the WDTx.</p>

## System Control Units (SCU)

**Table 273 Reset Values of WDTCPUySR (y=0-5)**

Reset Type	Reset Value	Note
Application Reset	FFFC 0010 <sub>H</sub>	WDTCPU0SR
Application Reset	FFFC 0008 <sub>H</sub>	WDTCPUySR (y=1-5)

### ENDINIT Global Control Register 0

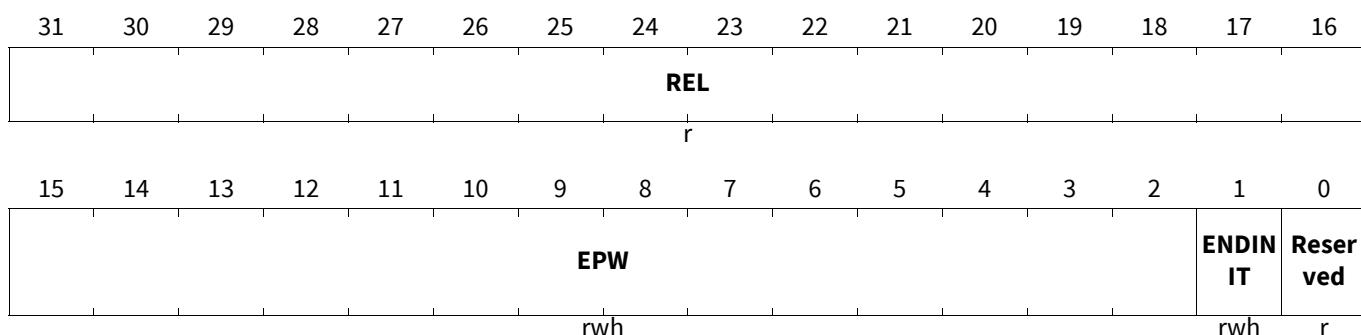
This register is part of the ENDINIT Timeout Counter. It is provided to allow an alternative way to access ENDINIT protected registers without affecting any CPU WDT counter values. The format of the EICON0 register is similar to that of a WDTxCON0 register, but the counter reload value is a fixed short timeout and the locking bit is the ENDINIT control. The password for ENDINIT access via EICON0 is static (i.e. No autosequencing is available) but may be updated on a Modify Access.

This allows an ENDINIT protected register to be accessed after a single Password Access write to EICON0, and to be re-locked with a single Modify Access write.

The System ENDINIT signal used to restrict register access is effectively the logical combination of EICON0.ENDINIT and all WDTxCON0.ENDINIT bits.

### EICON0

**ENDINIT Global Control Register 0** (029C<sub>H</sub>) **Application Reset Value: FFFC 000E<sub>H</sub>**



Field	Bits	Type	Description
<b>Reserved</b>	0	r	<b>Reserved Bit</b> Not writeable. Read as '0'.
<b>ENDINIT</b>	1	rwh	<b>End-of-Initialization Control Bit</b> The current value of ENDINIT is controlled by hardware. It is cleared after a valid EndInit Password Access to EICON0, and it is automatically set again after a valid EndInit Modify Access to EICON0. During a write to EICON0, the value written to this bit is only used for the password-protection mechanism and is not stored. This bit must be cleared during a Password Access to EICON0, and set during a Modify Access to EICON0. 0 <sub>B</sub> Access to Endinit-protected registers is permitted 1 <sub>B</sub> Access to Endinit-protected registers is not permitted unless one of WDTCPUyCON0.ENDINIT is 0.

## System Control Units (SCU)

Field	Bits	Type	Description
EPW	15:2	rwh	<p><b>User-Definable ENDINIT Password Field</b>  This bit field is written with an ENDINIT password value during a Modify Access. This password is independent from the CPU WDT passwords.  A read from this bitfield returns this password, but bits [7:2] are inverted (toggled) to ensure that a simple read/write is not sufficient to service the WDT.</p> <p>This bit field must be written with its current contents during a Password Access.</p> <p>The default ENDINIT password after Application Reset is <math>00000000111100_B</math></p>
REL	31:16	r	<p><b>Reload Value for the ENDINIT Timeout Counter</b>  The reload value for the ENDINIT Timeout Counter is fixed. This bitfield always reads as FFFCh and cannot be changed.</p> <p>This bit field must be written with its current contents during a Password Access. During a Modify Access this bitfield may contain any value and is ignored.</p>

### ENDINIT Global Control Register 1

The register EICON1 manages the ENDINIT Timeout Counter. It includes the disable request and frequency selection bits. EICON1 bits can only be modified when any ENDINIT=0.

#### EICON1

**ENDINIT Global Control Register 1** **(02A0<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
										IR1	0	DR	IR0	0	0	
											rw	r	rw	rw	r	r

Field	Bits	Type	Description
IR0	2	rw	<p><b>Input Frequency Request Control - IR1,IR0</b>  Bit IR0 and IR1 should be programmed together to determine the ENDINIT Timeout Counter frequency.  These bits can only be modified if system ENDINIT (E) is de-asserted. EISR.IS0 and EISR.IS1 are updated by these bits only when system ENDINIT (E) is re-asserted. As long as system ENDINIT (E) is de-asserted, EISR.IS0 and EISR.IS1 control the current input frequency of the ENDINIT Timeout Timer. When System ENDINIT (E) is re-asserted, EISR.IS0 and EISR.IS1 are updated with the new values of IR0 and IR1.</p> <p><math>0_B</math> If Bit IR1=0 Request to set input frequency to <math>f_{SPB}/16384</math>. Elseif Bit IR1=1 Request to set input frequency to <math>f_{SPB}/64</math>.</p> <p><math>1_B</math> If Bit IR1=0 Request to set input frequency to <math>f_{SPB}/256</math>. Elseif Bit IR1=1 Reserved. Do not use.</p>

## System Control Units (SCU)

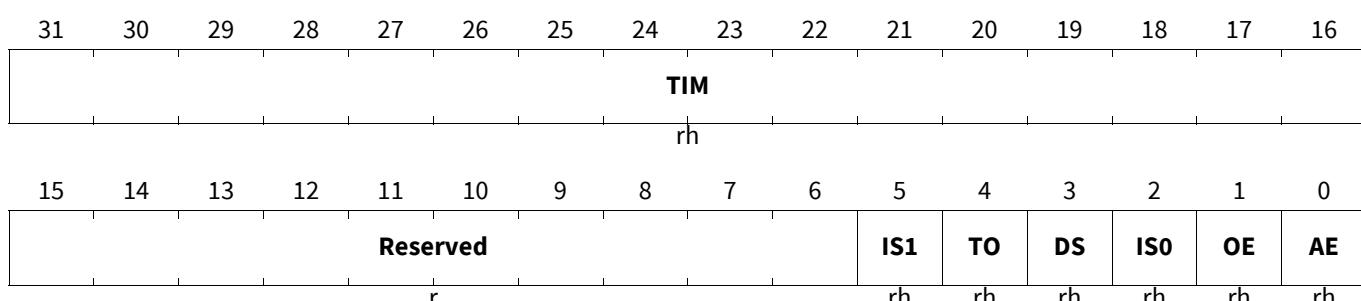
Field	Bits	Type	Description
DR	3	rw	<p><b>Disable Request Control Bit</b></p> <p>This bit can only be modified if the system ENDINIT (E) is de-asserted. EISR.DS is updated when system ENDINIT (E) is re-asserted. As long as system ENDINIT(E) is cleared, bit EISR.DS controls the current enable/disable status of the ENDINIT Timeout Counter. When system ENDINIT (E) is re-asserted, EISR.DS is updated with the state of DR.</p> <p>0<sub>B</sub> Request to enable the ENDINIT Timeout Counter 1<sub>B</sub> Request to disable the ENDINIT Timeout counter</p>
IR1	5	rw	<p><b>Input Frequency Request Control</b></p> <p>Bit IR0 and IR1 should be programmed together to determine the ENDINIT Timeout Counter frequency.</p> <p>These bits can only be modified if system ENDINIT (E) is de-asserted. EISR.IS0 and EISR.IS1 are updated by these bits only when system ENDINIT (E) is re-asserted. As long as system ENDINIT (E) is de-asserted, EISR.IS0 and EISR.IS1 control the current input frequency of the ENDINIT Timeout Timer. When System ENDINIT (E) is re-asserted, EISR.IS0 and EISR.IS1 are updated with the new values of IR0 and IR1.</p> <p>0<sub>B</sub> If Bit IR0=0 Request to set input frequency to <math>f_{SPB}/16384</math>. Elseif Bit IR0=1 Request to set input frequency to <math>f_{SPB}/256</math>. 1<sub>B</sub> If Bit IR0=0 Request to set input frequency to <math>f_{SPB}/64</math>. Elseif Bit IR0=1 Reserved. Do not use.</p>
0	0, 1, 4, 31:6	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

### ENDINIT Timeout Counter Status Register

The EISR register shows the current state of the ENDINIT Timeout Counter.

#### EISR

**ENDINIT Timeout Counter Status Register (02A4<sub>H</sub>) Application Reset Value: FFFC 0000<sub>H</sub>**



## System Control Units (SCU)

Field	Bits	Type	Description
<b>AE</b>	0	rh	<b>EICON0 Access Error Status Flag</b> This bit is set when an illegal Password Access or Modify Access to register EICON0 was attempted. This bit is only cleared on a valid EICON0.ENDINIT Modify Access $0_B$ No access error $1_B$ A access error has occurred
<b>OE</b>	1	rh	<b>EI Timeout Overflow Error Status Flag</b> This bit is set when EISR.TIM overflows from $FFFF_H$ to $FFFC_H$ . This bit is only cleared on a valid EICON0 Modify Access. $0_B$ No timeout overflow error $1_B$ A timeout overflow error has occurred
<b>IS0</b>	2	rh	<b>EI Timeout Input Clock Status - IS1,IS0</b> Bit IS0 and IS1 should be programmed together. These bits indicate the current ENDINIT Timeout Counter frequency. $0_B$ If Bit IS1=0 ENDINIT Timeout Counter frequency is $f_{SPB}/16384$ . Elseif Bit IS1=1 ENDINIT Timeout Counter frequency is $f_{SPB}/64$ . $1_B$ If Bit IS1=0 Request to set input frequency to $f_{SPB}/256$ . Elseif Bit IS1=1 Reserved. Do not use.
<b>DS</b>	3	rh	<b>EI Timeout Enable/Disable Status Flag</b> $0_B$ The ENDINIT Timeout Counter is enabled (After EICON0 Password Access) $1_B$ The ENDINIT Timeout Counter is disabled (After EICON0 Modify Access)
<b>TO</b>	4	rh	<b>EI Time-Out Mode Flag</b> $0_B$ The ENDINIT Timeout Counter is not operating in Time-Out Mode (After EICON0 Modify Access) $1_B$ The ENDINIT Timeout Counter is operating in Time-Out Mode (After EICON0 Password Access)
<b>IS1</b>	5	rh	<b>EI Timeout Input Clock Status</b> Bit IS0 and IS1 should be programmed together. These bits indicate the current ENDINIT Timeout Counter frequency. $0_B$ If Bit IS0=0 ENDINIT Timeout Counter frequency is $f_{SPB}/16384$ . Elseif Bit IS0=1 ENDINIT Timeout Counter frequency is $f_{SPB}/256$ . $1_B$ If Bit IS0=0 ENDINIT Timeout Counter frequency is $f_{SPB}/64$ . Elseif Bit IS0=1 Reserved. Do not use.
<b>Reserved</b>	15:6	r	<b>Reserved</b> These bits are unused and return '0' when read
<b>TIM</b>	31:16	rh	<b>Timer Value</b> Reflects the current content of the ENDINIT Timeout Counter.

## Safety ENDINIT Control Register 0

This register is part of the Safety ENDINIT Timeout Counter. It is provided to access Safety ENDINIT protected registers. The password for ENDINIT access via SEICON0 is static and may be updated on a Modify Access.

This allows a Safety ENDINIT protected register to be accessed after a single Password Access write to SEICON0, and to be re-locked with a single Modify Access write.

## System Control Units (SCU)

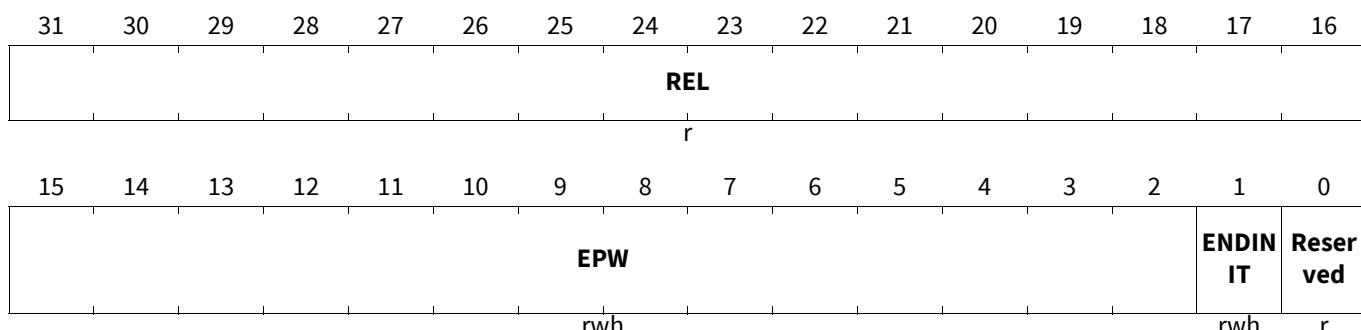
The Safety ENDINIT signal used to restrict register access is derived from the ENDINIT bit.

### SEICON0

#### Safety ENDINIT Control Register 0

(02B4<sub>H</sub>)

Application Reset Value: FFFC 000E<sub>H</sub>



Field	Bits	Type	Description
Reserved	0	r	<b>Reserved Bit</b> Not writeable. Read as '0'.
ENDINIT	1	rwh	<b>End-of-Initialization Control Bit</b> The current value of ENDINIT is controlled by hardware. It is cleared after a valid EndInit Password Access to SEICON0, and it is automatically set again after a valid EndInit Modify Access to SEICON0. During a write to SEICON0, the value written to this bit is only used for the password-protection mechanism and is not stored. This bit must be cleared during a Password Access to SEICON0, and set during a Modify Access to SEICON0. 0 <sub>B</sub> Access to Safety Endinit-protected registers is permitted 1 <sub>B</sub> Access to Safety Endinit-protected registers is not permitted unless WDTSCON0.ENDINIT is 0.
EPW	15:2	rwh	<b>User-Definable Safety ENDINIT Password Field</b> This bit field is written with an ENDINIT password value during a Modify Access. This password is independent from the CPU WDT or WDTS passwords. A read from this bitfield returns this password, but bits [7:2] are inverted (toggled) to ensure that a simple read/write is not sufficient to service the Safety ENDINIT Timeout Counter. This bit field must be written with its current contents during a Password Access. The default ENDINIT password after Application Reset is 00000000111100 <sub>B</sub>
REL	31:16	r	<b>Reload Value for the Safety ENDINIT Timeout Counter</b> The reload value for the Safety ENDINIT Timeout Counter is fixed. This bitfield always reads as FFFCh and cannot be changed. This bit field must be written with its current contents during a Password Access. During a Modify Access this bitfield may contain any value and is ignored.

## System Control Units (SCU)

### Safety ENDINIT Control Register 1

The register SEICON1 manages the Safety ENDINIT Timeout Counter. It includes the disable request and frequency selection bits. SEICON1 bits can only be modified when SEICON0.ENDINIT=0.

#### SEICON1

#### Safety ENDINIT Control Register 1 (02B8<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															

Field	Bits	Type	Description
IR0	2	rw	<b>Input Frequency Request Control - IR1,IR0</b> Bit IR0 and IR1 should be programmed together to determine the Safety ENDINIT Timeout Counter frequency. These bits can only be modified when Safety ENDINIT (SE) is de-asserted. SEISR.IS0 and SEISR.IS1 are updated by these bits only when Safety ENDINIT (SE) is re-asserted. As long as an Safety ENDINIT is cleared, SEISR.IS0 and SEISR.IS1 control the current input frequency of the Safety ENDINIT Timeout Timer. When Safety ENDINIT(SE) is re-asserted, SEISR.IS0 and SEISR.IS1 are updated with the new values of IR0 and IR1. $0_B$ If Bit IR1=0 Request to set input frequency to $f_{SPB}/16384$ . Elseif Bit IR1=1 Request to set input frequency to $f_{SPB}/64$ . $1_B$ If Bit IR1=0 Request to set input frequency to $f_{SPB}/256$ . Elseif Bit IR1=1 Reserved. Do not use
DR	3	rw	<b>Disable Request Control Bit</b> This bit can only be modified when Safety ENDINIT (SE) is de-asserted. SEISR.DS is updated when Safety ENDINIT is re-asserted. As long as Safety ENDINIT is deasserted, bit SEISR.DS controls the current enable/disable status of the WDT. When Safety ENDINIT is re-asserted, SEISR.DS is updated with the state of DR. $0_B$ Request to enable the Safety ENDINIT Timeout counter $1_B$ Request to disable the Safety ENDINIT Timeout counter

## System Control Units (SCU)

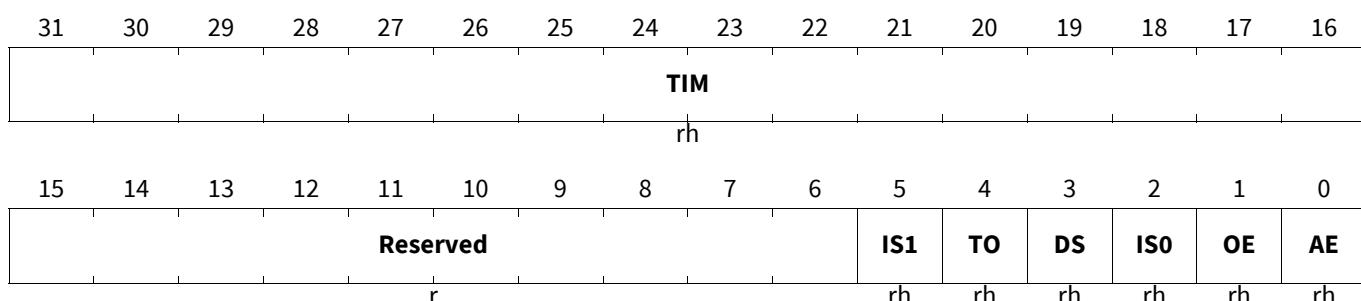
Field	Bits	Type	Description
IR1	5	rw	<p><b>Input Frequency Request Control</b></p> <p>Bit IR0 and IR1 should be programmed together to determine the Safety ENDINIT Timeout Counter frequency.</p> <p>These bits can only be modified when Safety ENDINIT (SE) is de-asserted. SEISR.IS0 and SEISR.IS1 are updated by these bit only when Safety ENDINIT (SE) is re-asserted. As long as an ENDINIT is cleared, SEISR.IS0 and SEISR.IS1 control the current input frequency of the ENDINIT Timeout Counter. When Safety ENDINIT(SE) is re-asserted, SEISR.IS0 and SEISR.IS1 is updated with the new values of IR0 and IR1.</p> <p>0<sub>B</sub> If Bit IR0=0 Request to set input frequency to <math>f_{SPB}/16384</math>. Elseif Bit IR0=1 Request to set input frequency to <math>f_{SPB}/256</math>.</p> <p>1<sub>B</sub> If Bit IR0=0 Request to set input frequency to <math>f_{SPB}/64</math>. Elseif Bit IR0=1 Reserved. Do not use.</p>
0	0, 1, 4, 31:6	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

### Safety ENDINIT Timeout Status Register

The SEISR register shows the current state of the Safety ENDINIT Timeout Counter.

#### SEISR

**Safety ENDINIT Timeout Status Register (02BC<sub>H</sub>) Application Reset Value: FFFC 0000<sub>H</sub>**



Field	Bits	Type	Description
AE	0	rh	<p><b>SEICON0 Access Error Status Flag</b></p> <p>This bit is set when an illegal Password Access or Modify Access to register SEICON0 was attempted. This bit is only cleared on a valid SEICON0.ENDINIT Modify Access</p> <p>0<sub>B</sub> No access error</p> <p>1<sub>B</sub> An access error has occurred</p>
OE	1	rh	<p><b>SEI Timeout Overflow Error Status Flag</b></p> <p>This bit is set when SEISR.TIM overflows from FFFF<sub>H</sub> to FFFC<sub>H</sub>. This bit is only cleared on a valid SEICON0 Modify Access.</p> <p>0<sub>B</sub> No overflow error</p> <p>1<sub>B</sub> An overflow error has occurred</p>

## System Control Units (SCU)

Field	Bits	Type	Description
IS0	2	rh	<p><b>SEI Timeout Input Clock Status</b></p> <p>Bit IS0 and IS1 should be programmed together. These bits indicate the current Safety ENDINIT Timeout Counter clock period. They are updated with the state of bits SEICON1.IR0 and SEICON1.IR1 after a valid SEICON0 Modify Access.</p> <p>0<sub>B</sub> If Bit IS1=0 Safety ENDINIT Timeout Counter frequency is <math>f_{SPB}/16384</math>. Elseif Bit IS1=1 Safety ENDINIT Timeout Counter frequency is <math>f_{SPB}/64</math>.</p> <p>1<sub>B</sub> If Bit IS1=0 Request to set input frequency to <math>f_{SPB}/256</math>. Elseif Bit IS1=1 Reserved. Do not use.</p>
DS	3	rh	<p><b>SEI Enable/Disable Status Flag</b></p> <p>0<sub>B</sub> The SEI Timeout Counter is enabled (After SEICON0 Password Access)</p> <p>1<sub>B</sub> The SEI timeout counter is disabled (After SEICON0 Modify Access)</p>
TO	4	rh	<p><b>SEI Time-Out Mode Flag</b></p> <p>0<sub>B</sub> The SEI Timeout Counter is not operating in Time-Out Mode (After SEICON0 Modify Access)</p> <p>1<sub>B</sub> The SEI timeout counter is operating in Time-Out Mode (After SEICON0 Password Access)</p>
IS1	5	rh	<p><b>SEI Timeout Input Clock Status - IS0, IS1</b></p> <p>Bit IS0 and IS1 should be programmed together. These bits indicate the current Safety ENDINIT Timeout Counter clock period . They are updated with the state of bits SEICON1.IR0 and SEICON1.IR1 after a valid SEICON0 Modify Access.</p> <p>0<sub>B</sub> If Bit IS0=0 Safety ENDINIT Timeout Counter frequency is <math>f_{SPB}/16384</math>. Elseif Bit IS0=1 Safety ENDINIT Timeout Counter frequency is <math>f_{SPB}/256</math>.</p> <p>1<sub>B</sub> If Bit IS0=0 Safety ENDINIT Timeout Counter frequency is <math>f_{SPB}/64</math>. Elseif Bit IS0=1 Reserved. Do not use.</p>
Reserved	15:6	r	<p><b>Reserved</b></p> <p>These bits are unused and return '0' when read</p>
TIM	31:16	rh	<p><b>Timer Value</b></p> <p>Reflects the current content of the Safety EINDINIT Timeout Counter.</p>

## 9.5 External Request Unit (ERU)

The External Request Unit (ERU) is a versatile event and pattern detection unit. Its major task is the **generation of interrupts based on selectable trigger events** (e.g. to generate external interrupt requests if an edge occurs at an input pin).

The detected events can also be used by other target modules to trigger or to gate module-specific actions.

The available trigger sources and defined target modules are defined in the SCU Appendix for each device under the headline "Connectivity".

### 9.5.1 Feature List

- Supports generation of interrupts based on selectable trigger events at different inputs

## System Control Units (SCU)

- 8 independent Input Channels for input selection and conditioning of trigger or gating functions.
- Event distribution with a Connecting Matrix which defines the events of the Input Channel x that lead to a reaction of an Output Channel y.
- 8 independent Output Channels for combination of events, definition of their effects and distribution to the system (e.g. interrupt generation, timer triggering ...)

### 9.5.1.1 Delta to AURIX

The most significant changes between the TC2xx ERU and TC3xx ERU are:

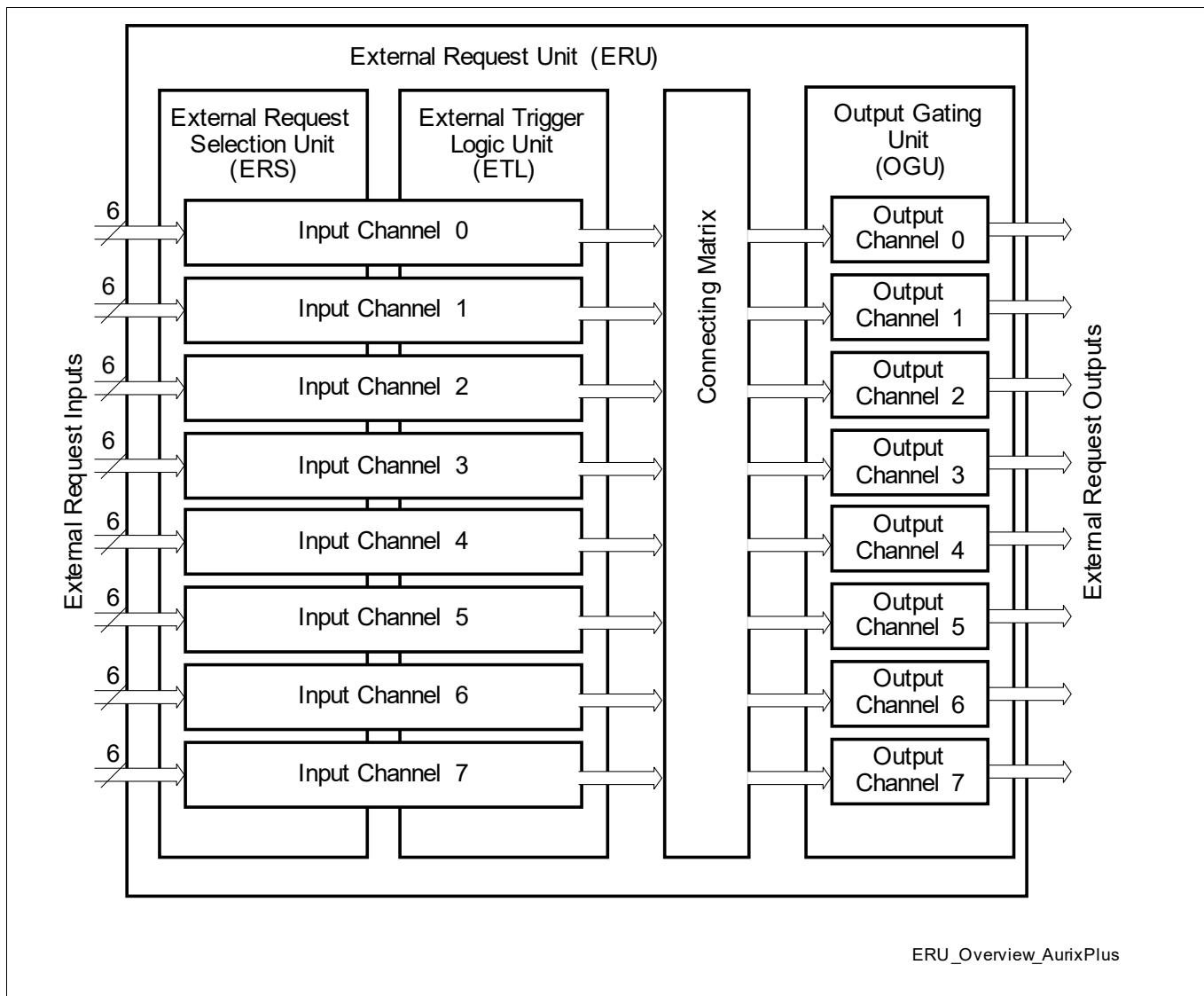
- Some register address changes
- ERU input mux widened from 4 to 6 inputs per channel to accommodate additional trigger sources
- Programmable digital glitch filtering available on ERU REQx inputs
- Ability to generate SMU alarms from ERU

### 9.5.2 Introduction

The ERU can be split in three main functional parts:

- **Input Channels x**
- **Connecting Matrix**
- **Output Channels y**

## System Control Units (SCU)



**Figure 74 External Request Unit Overview**

These tasks are handled by the following building blocks:

- An **External Request Select Unit (ERSx)** per Input Channel allows the selection of one input vector out of the 6 possible available inputs.
- An **Event Trigger Logic (ETLx)** per Input Channel allows the definition of the transition (edge selection, or by software) that lead to a trigger event and can also store this status. Here, the input levels of the selected signals are translated into events (event detected = event flag becomes set, independent of the polarity of the original input signals).
- The **Connecting Matrix** distributes the events and status flags generated by the Input Channels to the Output Channels.
- An Output Gating Unit (OGUy) per Output Channel that combines the available trigger events and status information from the Input Channels. An event of one Input Channel can lead to reactions of several Output Channels, or also events of several Input Channels can be combined to a reaction of one Output Channel (pattern detection).

Different types of reactions can be configured, e.g. generation of interrupts.

The inputs to the ERU can be selected from a large number of input signals. 16 of these inputs come directly from input ports, but other inputs come from various peripheral module status signals. Usually, such inputs would be

## System Control Units (SCU)

selected for an ERU function when the module input function is not used by the application, or when the module is not used at all. However, it is also possible to select an input which is also used by the other module, to also be used in the ERU as a trigger or to be combined with other signals (e.g. to generate an interrupt trigger when a start-of-frame is detected on a selected communication interface input) - to know the device specific input and output connections, please address the appendix file.

### 9.5.3 REQxy Digital PORT Input Glitch Filter (FILT)

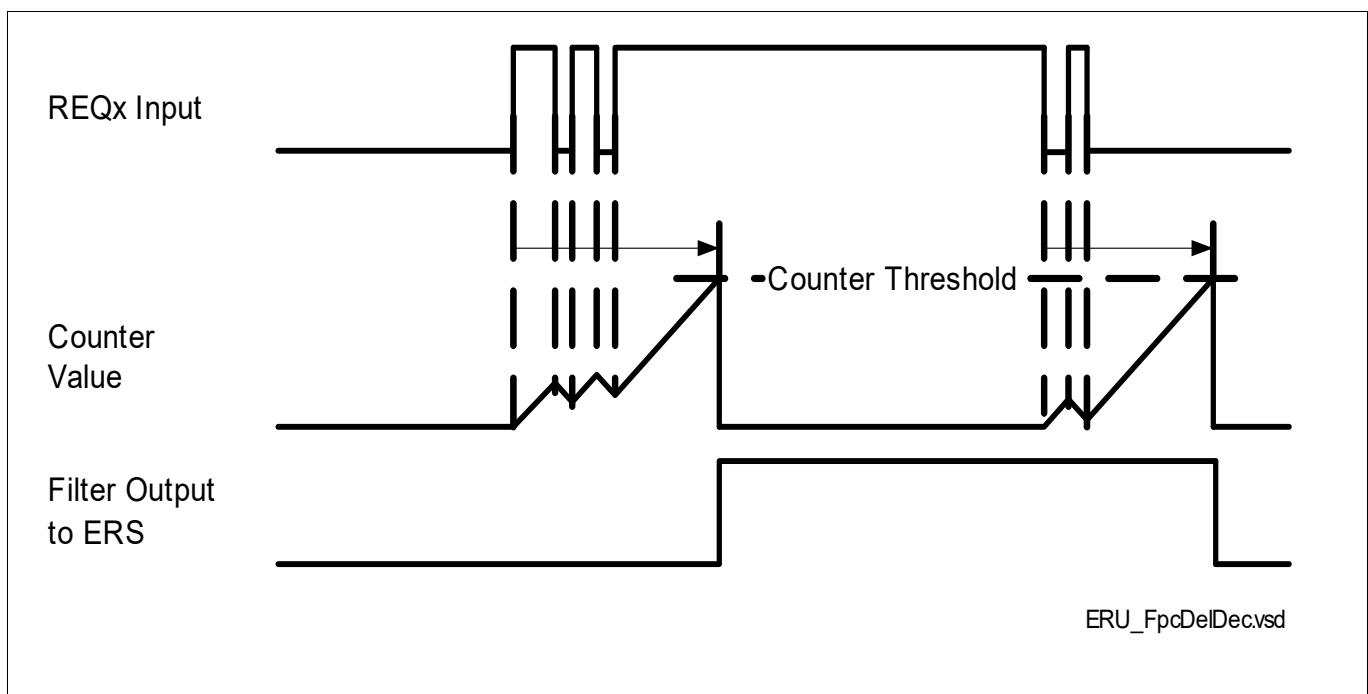
Signal noise can lead to unwanted fast transitions on input pins from PORTS. These unwanted transitions may be suppressed by digital glitch filters similar to Filter and Prescaler Cells (FPC) in Delayed Debounce Filter Mode with up and down (no reset).

The filter calculates the integral of the signal. If the integral reaches a programmable saturation point, the input change is passed on to the ERS.

The REQxy glitch filters are only available on the REQxy inputs coming directly from PORTS. They are enabled and configured using register **EIFILT**. The filters sample the input and are clocked with a clock of frequency  $f_{FILT}$ , where  $f_{FILT} = f_{SPB}/\text{EIFILT}.FILTDIV$ . On each sample period, if the state of the input sample differs from the current state of the filter output, then an internal counter is incremented by one. When the counter matches the compare threshold value stored in **EIFILT.DEPTH**, the state of the filter output is inverted and the counter is reset to zero. When the state of the input sample matches the current state of the filter output and the counter is not zero, the counter is decremented by one.

The filter predivider can be programmed to a value between 1 and 15, giving a range of possible glitch characteristics from 10ns to  $> 2\mu s$

The depth of the filter can be programmed to a value between 1 and 15. Typically a depth of 3 to 5  $T_{filt}$  is sufficient. By default it is cleared. If DEPTH is cleared all filters are inactive.

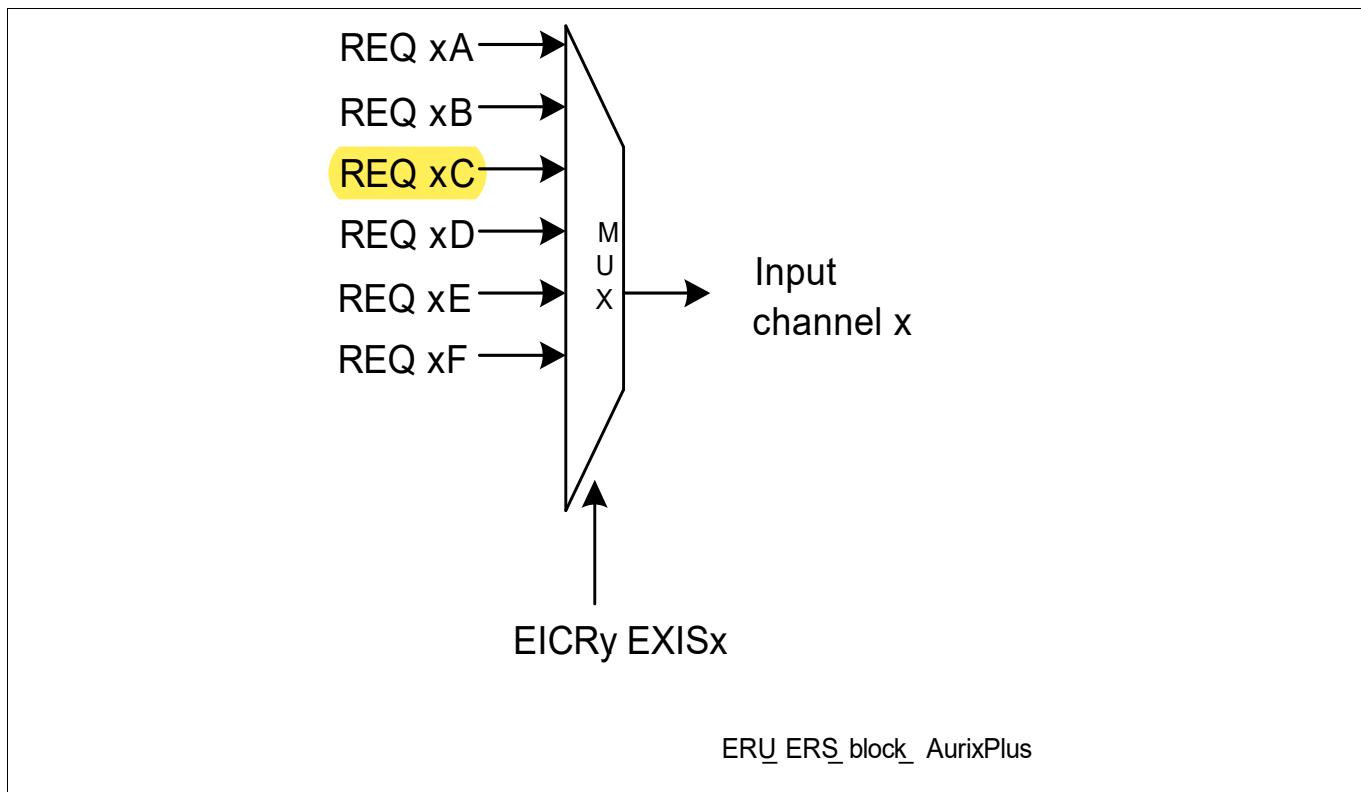


**Figure 75** REQxy Digital Filter

### 9.5.4 External Request Selector Unit (ERS)

Each ERS selects one of six inputs as the one input signal of the respective input channel. **Figure 76** shows the structure of this block.

## System Control Units (SCU)



**Figure 76 External Request Select Unit Overview**

The ERS unit for channel x is controlled via bit field EICRy.EXISx.

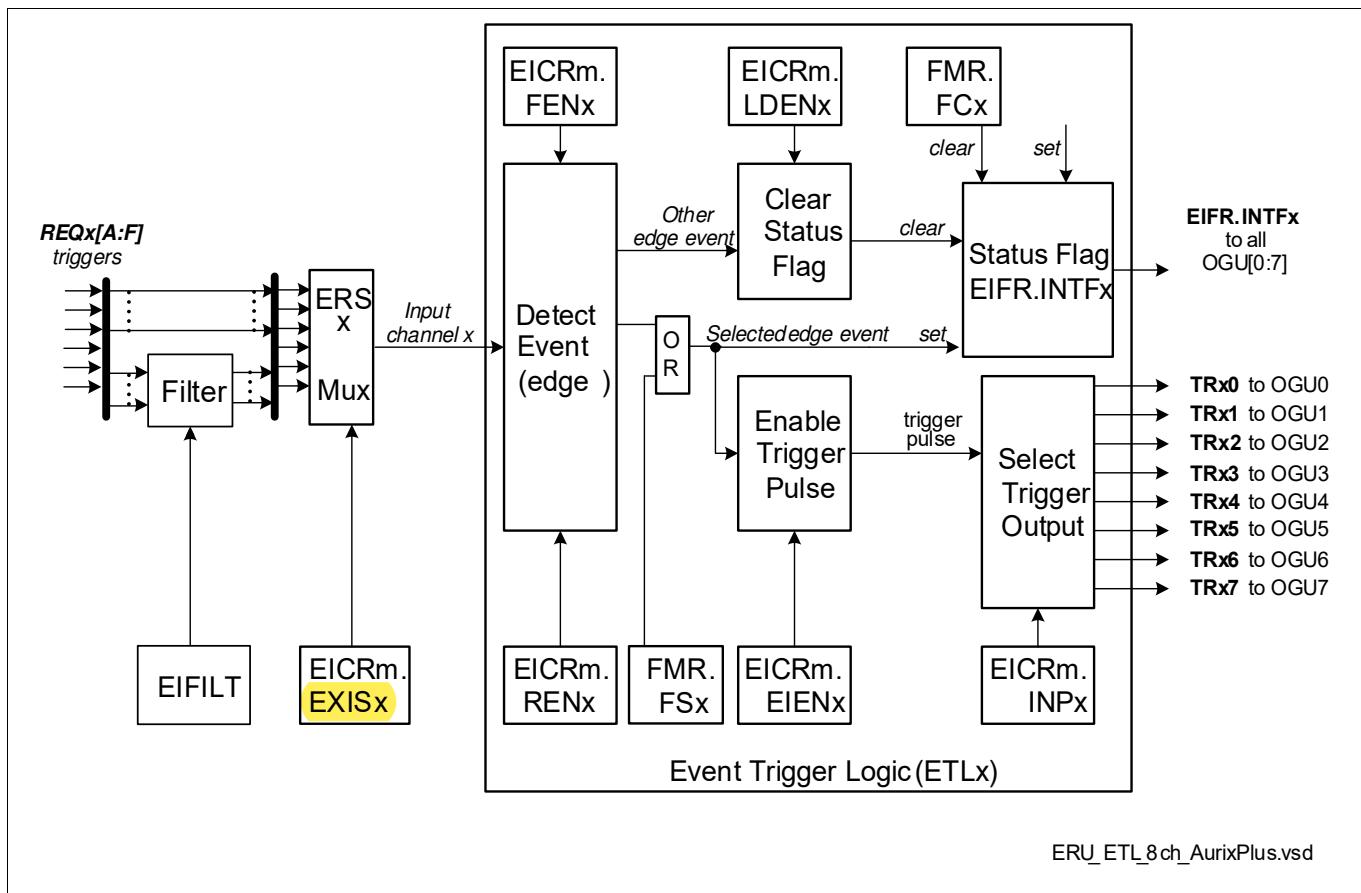
### 9.5.5 Event Trigger Logic (ETL)

For each Input Channel x, an event trigger logic ETLx derives a trigger event and a status from the input channel x delivered by the associated ERSx unit. Each ETLx is based on an edge detection block, where the detection of a rising or a falling edge can be individually enabled. Both edges lead to a trigger event if both enable bits are set (e.g. to handle a toggling input).

Each pair of the four ETL units has an associated EICRy register, that controls all options of an ETL (the register also holds control bits for the associated ERS unit pair).

An overview of the Event Trigger Logic block is shown below.

## System Control Units (SCU)



**Figure 77 Event Trigger Logic Overview**

When the selected event (edge) is detected, the status flag EIFR.INTFx becomes set.

The status flag is cleared automatically if the “opposite” event is detected, if so enabled via bit EICRy.LDENx = 1. For example, if only the falling edge detection is enabled to set the status flag, it is cleared when the rising edge is detected. In this mode, it can be used for pattern detection where the actual status of the input is important (enabling both edge detections is not useful in this mode).

The output of the status flag is connected to all following Output Gating Units (OGUz) in parallel (see **Figure 78**) to provide **pattern detection capability of all OGUz** units based on different or the same status flags.

In addition to the modification of the status flag, a trigger pulse output TRxz of ETLx can be enabled (by bit EICRy.EIENx) and selected to **trigger actions in one of the OGUz** units. The target OGUz for the trigger is selected by bit field EICRy.INPx.

The trigger becomes active when the selected edge event is detected, independently from the status flag EIFR.INTFx.

## System Control Units (SCU)

### 9.5.6 Connecting Matrix

The connecting matrix distributes the trigger signals (TRxy) and status signals (EIFR.INPFx) from the different ETLx units between the OGUs. **Figure 78** provides a complete overview of the connections between the ETLx and the OGUs.

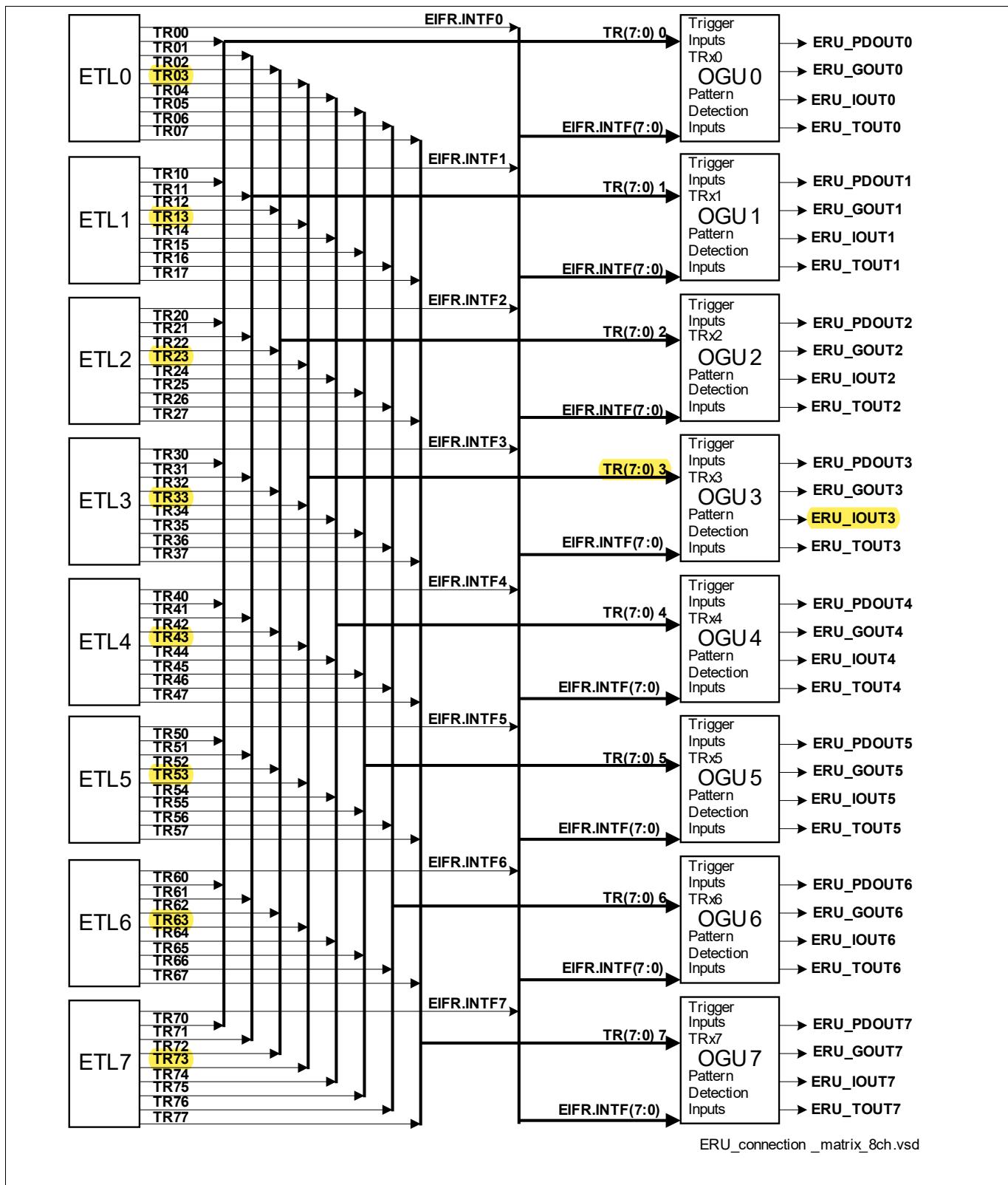


Figure 78 Connecting Matrix between ETLx and OGUs

## System Control Units (SCU)

### 9.5.7 Output Gating Unit (OGU)

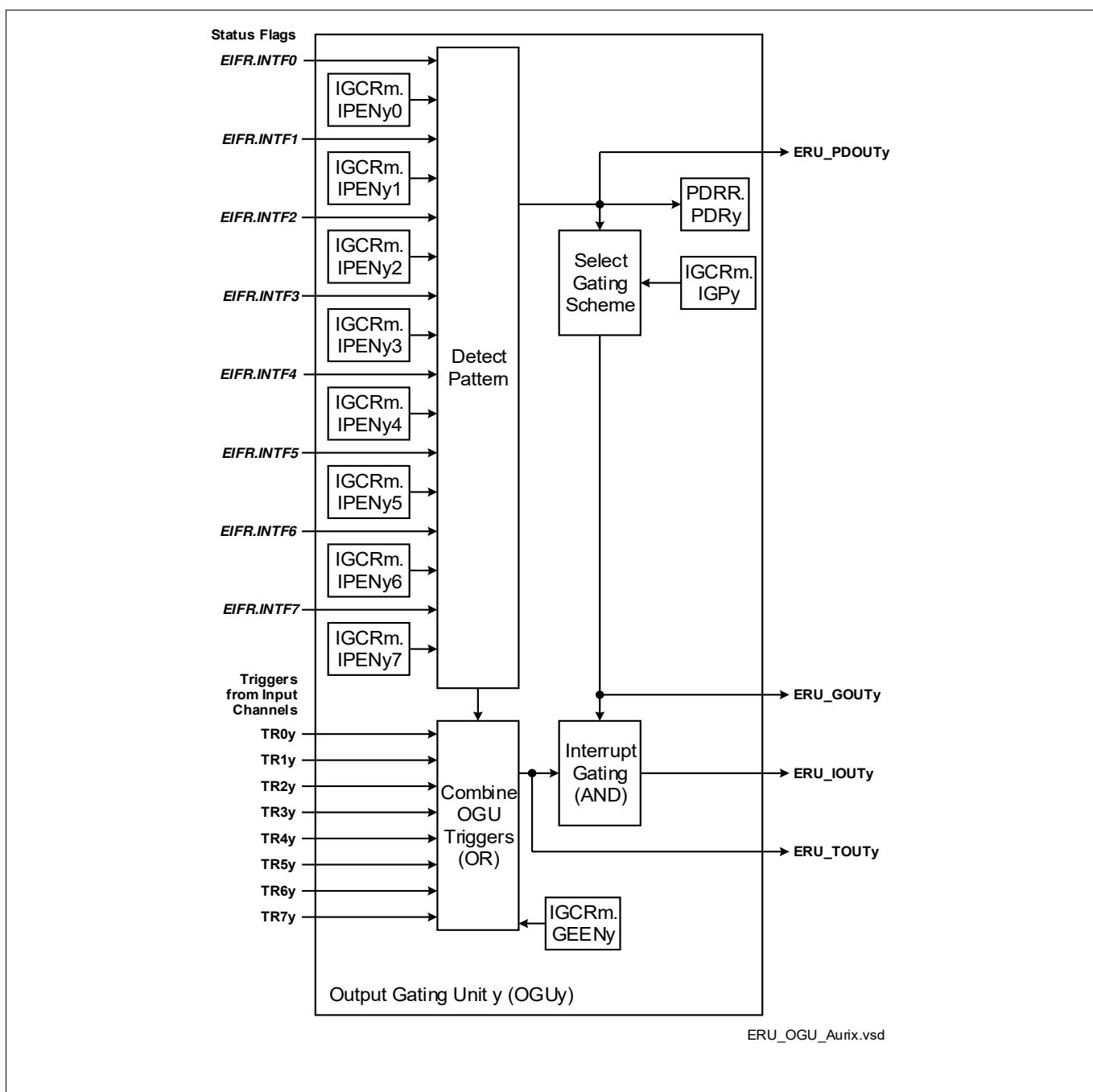
Each OGUy unit combines the available trigger events and status flags from the Input Channels and distributes the results to the system. **Figure 79** illustrates the logic blocks within an OGUy unit. All functions of an OGUy unit are controlled by the associated IGCRm registers, one for each pair of output channels e.g. IGCR1 for OGU2 and OGU3. The function of an OGUy unit can be split into two parts:

- Trigger combination:**

All trigger signals TRxy from the Input Channels that are enabled and directed to OGUy and a pattern change event (if enabled) are logically OR-combined.

- Pattern detection:**

The status flags EIFR.INTFx of the Input Channels can be enabled to take part in the pattern detection. A pattern match is detected while all enabled status flags are set.



**Figure 79 Output Gating Unit for Output Channel y**

## System Control Units (SCU)

Each OGUY units generates 4 output signals that are distributed to the system.

- **ERU\_PDOUTy** to directly output the pattern match information for gating purposes in defined target modules (pattern match = 1).
- **ERU\_GOUTy** to output the pattern match or pattern miss information (inverted pattern match), or a permanent 0 or 1 under software control for gating purposes in other modules.
- **ERU\_TOUTy** as combination of a peripheral trigger, a pattern detection result change event, or the ETLx trigger outputs TRxy to trigger actions in other modules.
- **ERU\_IOUTy** as gated trigger output (ERU\_GOUTy logical AND-combined with ERU\_TOUTy) to trigger interrupts (e.g. the interrupt generation can be gated to allow interrupt activation during a certain time window).

### 9.5.7.1 Trigger Combination

The “Combine OGU Triggers” block logically OR-combines different trigger inputs to form a common trigger ERU\_TOUTy. Possible trigger inputs are:

- In each ETLx unit of the **Input Channels**, the trigger output TRxy can be enabled and the trigger event can be directed to one of the OGUY units.
- In the case that at least one **pattern detection** input is enabled (IGCRm.IPENxy) and a change of the pattern detection result from pattern match to pattern miss (or vice-versa) is detected, a trigger event is generated to indicate a pattern detection result event (if enabled by IGCRm.GEENy).

The trigger combination offers the possibility to program different trigger criteria for several input signals (independently for each Input Channel) or peripheral signals, and to combine their effects to a single output, e.g. to generate an interrupt or to start an ADC conversion. This combination capability allows the generation of an interrupt per OGU that can be triggered by several inputs (multitude of request sources -> one reaction).

### 9.5.7.2 Pattern Detection

The “Detect Pattern” block allows the combination of the status flags of all ETLx units. Each status flag can be individually included or excluded from the pattern detection for each OGUY, via control bits IGCRm.IPENxy. The pattern detection block outputs the following pattern detection results:

- **Pattern match** (PDRR.PDRy = 1 and ERU\_PDOUTy = 1):  
A pattern match is indicated while all status flags that are included in the pattern detection are 1.
- **Pattern miss** (PDRR.PDRy = 0 and ERU\_PDOUTy = 0):  
A pattern miss is indicated while at least one of the status flags that are included in the pattern detection is 0.

In addition, the pattern detection can deliver a trigger event if the pattern detection result changes from match to miss or vice-versa (if enabled by IGCRm.GEENy = 1). The pattern result change event is logically OR-combined with the other enabled trigger events to support interrupt generation or to trigger other module functions (e.g. in an ADC). The event is indicated when the pattern detection result changes and PDRR.PDRy becomes updated.

The interrupt generation in the OGUY is based on the trigger ERU\_TOUTy that can be gated (masked) with the pattern detection result ERU\_PDOUTy. This allows an automatic and reproducible generation of interrupts during a certain time window, where the request event is elaborated by the trigger combination block and the time window information (gating) is given by the pattern detection. For example, interrupts can be issued on a regular time base while a combination of input signals occurs (pattern detection based on ETLx status bits).

Interrupt/service requests can be generated only by the ERU OGU[0-3] via its outputs ERU\_IOUT[0-3]. These outputs are connected to the IR Service Request registers SRC\_SCUERU[0-3] via the signals ERU\_INT[0-3] as described below.

**System Control Units (SCU)****Table 274 OGU to SRC connection**

<b>OGUy.ERU_IOUTy (OGU I-output signal)</b>	<b>SRC_SCUERUx (interrupt SRC register)</b>
OGU0.ERU_IOUT0	SRC_SCUERU0
OGU1.ERU_IOUT1	SRC_SCUERU1
OGU2.ERU_IOUT2	SRC_SCUERU2
<b>OGU3.ERU_IOUT3</b>	<b>SRC_SCUERU3</b>

The mapping of interrupt requests to OGUy blocks is shown in the ERU connection diagram.

A programmable gating scheme introduces flexibility to adapt to application requirements and allows the generation of interrupt requests ERU\_IOUTy under different conditions:

- **Pattern match** ( $IGCRm.IGPy = 10_B$ ):  
An interrupt request is issued when a trigger event occurs while the pattern detection shows a pattern match.
- **Pattern miss** ( $IGCRm.IGPy = 11_B$ ):  
An interrupt request is issued when the trigger event occurs while the pattern detection shows a pattern miss.
- **Independent** of pattern detection ( $IGCRm.IGPy = 01_B$ ):  
In this mode, each occurring trigger event leads to an interrupt request. The pattern detection output can be used independently from the trigger combination for gating purposes of other peripherals (independent use of ERU\_TOUTy and ERU\_PDOUTy with interrupt requests on trigger events).
- **No interrupts** ( $IGCRm.IGPy = 00_B$ , default setting)  
In this mode, an occurring trigger event does not lead to an interrupt request. The pattern detection output can be used independently from the trigger combination for gating purposes of other peripherals (independent use of ERU\_TOUTy and ERU\_PDOUTy without interrupt requests on trigger events).

### 9.5.7.3 Triggering SMU alarms

It is possible to trigger alarms to the SMU via the functional outputs of the ERU.

This is normally used to trigger a safety alarm from an external component, e.g. the component fail signal is connected to the ERU inputs and when it fails the ERU detects a transition on this input, it will trigger an alarm to the SMU.

The ERU outputs that are connected to the SMU are the IOUT signals - IOUT0 throughout IOUT7.

Please address the SMU chapter to know which alarm lines are being used for each signal.

To enable the triggers on the ERU, one shall do the following steps:

1. Identifiy which ERU input signal shall be used as alarm trigger source
2. Configure then , the specific ETL unit by:
  - configure the EXISx register field to select the input
  - configure the RENx and FENx register fields to select if the input signal is active on a rising edge or falling edge
  - if the trigger signal from the ETL shall be used instead of the status flag, as source for the Alarm, then configure the INPx field
3. Configure the specific OGU that shall be used to trigger the SMU Alarm - this is linked to which output shall be used, e.g. IOUT4, then the OGU4 shall be configured:
  - configure the IPENy0 to IPENy7 if one or more status flags are used to generate or gate the trigger signal
  - configure GENNy field to use the pattern detection path to trigger/generate the Alarm/IOUT
  - configure the IGPy field to select the gating scheme for the output Alarm/IOUT

---

**System Control Units (SCU)**

One shall address the SMU chapter, to understand if additional configuration is needed to enable a specific alarm line/trigger.

## System Control Units (SCU)

### 9.5.8 External Request Unit Registers

#### External Input Filter Register

The External Input Filter Register enables and configures optional digital Glitch Filters on the ERS inputs from PORTS.

#### EIFILT

**External Input Filter Register** **(020C<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
																FILRQ 7C
																rw
DEPTH					FILTDIV											
																r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
FILRQ 3B	FILRQ 2B	FILRQ 4D	FILRQ 6D	FILRQ 7A	FILRQ 1A	FILRQ 6A	FILRQ 4A	FILRQ 2C	FILRQ 3C	FILRQ 1C	FILRQ 0C	FILRQ 3A	FILRQ 2A	FILRQ 5A	FILRQ 0A	
rw																

Field	Bits	Type	Description
<b>FILRQ0A</b>	0	rw	<b>Filter Enable for REQ0A</b> 0 <sub>B</sub> REQ0A is unfiltered 1 <sub>B</sub> REQ0A glitch filter is enabled
<b>FILRQ5A</b>	1	rw	<b>Filter Enable for REQ5A</b> 0 <sub>B</sub> REQ5A is unfiltered 1 <sub>B</sub> REQ5A glitch filter is enabled
<b>FILRQ2A</b>	2	rw	<b>Filter Enable for REQ2A</b> 0 <sub>B</sub> REQ2A is unfiltered 1 <sub>B</sub> REQ2A glitch filter is enabled
<b>FILRQ3A</b>	3	rw	<b>Filter Enable for REQ3A</b> 0 <sub>B</sub> REQ3A is unfiltered 1 <sub>B</sub> REQ3A glitch filter is enabled
<b>FILRQ0C</b>	4	rw	<b>Filter Enable for REQ0C</b> 0 <sub>B</sub> REQ0C is unfiltered 1 <sub>B</sub> REQ0C glitch filter is enabled
<b>FILRQ1C</b>	5	rw	<b>Filter Enable for REQ1C</b> 0 <sub>B</sub> REQ1C is unfiltered 1 <sub>B</sub> REQ1C glitch filter is enabled
<b>FILRQ3C</b>	6	rw	<b>Filter Enable for REQ3C</b> 0 <sub>B</sub> REQ3C is unfiltered 1 <sub>B</sub> REQ3C glitch filter is enabled
<b>FILRQ2C</b>	7	rw	<b>Filter Enable for REQ2C</b> 0 <sub>B</sub> REQ2C is unfiltered 1 <sub>B</sub> REQ2C glitch filter is enabled

**System Control Units (SCU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>FILRQ4A</b>	8	rw	<b>Filter Enable for REQ4A</b> 0 <sub>B</sub> REQ4A is unfiltered 1 <sub>B</sub> REQ4A glitch filter is enabled
<b>FILRQ6A</b>	9	rw	<b>Filter Enable for REQ6A</b> 0 <sub>B</sub> REQ6A is unfiltered 1 <sub>B</sub> REQ6A glitch filter is enabled
<b>FILRQ1A</b>	10	rw	<b>Filter Enable for REQ1A</b> 0 <sub>B</sub> REQ1A is unfiltered 1 <sub>B</sub> REQ1A glitch filter is enabled
<b>FILRQ7A</b>	11	rw	<b>Filter Enable for REQ7A</b> 0 <sub>B</sub> REQ7A is unfiltered 1 <sub>B</sub> REQ7A glitch filter is enabled
<b>FILRQ6D</b>	12	rw	<b>Filter Enable for REQ6D</b> 0 <sub>B</sub> REQ6D is unfiltered 1 <sub>B</sub> REQ6D glitch filter is enabled
<b>FILRQ4D</b>	13	rw	<b>Filter Enable for REQ4D</b> 0 <sub>B</sub> REQ4D is unfiltered 1 <sub>B</sub> REQ4D glitch filter is enabled
<b>FILRQ2B</b>	14	rw	<b>Filter Enable for REQ2B</b> 0 <sub>B</sub> REQ2B is unfiltered 1 <sub>B</sub> REQ2B glitch filter is enabled
<b>FILRQ3B</b>	15	rw	<b>Filter Enable for REQ3B</b> 0 <sub>B</sub> REQ3B is unfiltered 1 <sub>B</sub> REQ3B glitch filter is enabled
<b>FILRQ7C</b>	16	rw	<b>Filter Enable for REQ7C</b> 0 <sub>B</sub> REQ7C is unfiltered 1 <sub>B</sub> REQ7C glitch filter is enabled
<b>FILTDIV</b>	27:24	rw	<b>Digital Glitch Filter Clock Predivider</b> This field controls a predivider to generate the digital filter sample clock $T_{filt} = T_{spb} * FILTDIV$ A value of zero in this register disables all glitch filtering.
<b>DEPTH</b>	31:28	rw	<b>Digital Glitch Filter Depth</b> DEPTH determines the number of port input samples considered in the calculation of the floating average digital filter output for all enabled FLRQ filters. A value of zero in this register disables all glitch filtering.
<b>0</b>	23:17	r	<b>Reserved</b> Read as 0; should be written with 0.

**External Input Channel Register i**

Each External Input Channel Register EICR<sub>i</sub> (i=0 to 3) contains bits to configure the external request selection ERS and the event trigger logic ETL for two input channels.

## System Control Units (SCU)

EICR<sub>i</sub> (i=0-3)

External Input Channel Register i

(0210<sub>H</sub>+i\*4)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		INP1		EIEN1	LDEN1	REN1	FEN1	0		EXIS1			0		0
r		rw		rw	rw	rw	rw	r		rw			r		r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		INPO		EIENO	LDENO	RENO	FENO	0		EXISO			0		0
r		rw		rw	rw	rw	rw	r		rw			r		r

Field	Bits	Type	Description
EXISO	6:4	rw	<b>External Input Selection 0</b> This bit field determines which input line is selected for Input Channel (2i). 000 <sub>B</sub> Input (2i) A is selected 001 <sub>B</sub> Input (2i) B is selected <b>010<sub>B</sub> Input (2i) C is selected</b> 011 <sub>B</sub> Input (2i) D is selected 100 <sub>B</sub> Input (2i) E is selected 101 <sub>B</sub> Input (2i) F is selected 110 <sub>B</sub> Reserved 111 <sub>B</sub> Reserved
FENO	8	rw	<b>Falling Edge Enable 0</b> This bit determines if the falling edge of Input Channel (2i) is used to set bit INTF(2i). 0 <sub>B</sub> The falling edge is not used 1 <sub>B</sub> The detection of a falling edge of Input Channel 0 generates a trigger event. INTF(2i) becomes set.
RENO	9	rw	<b>Rising Edge Enable 0</b> This bit determines if the rising edge of Input Channel (2*i) is used to set bit INTF(2i). 0 <sub>B</sub> The rising edge is not used 1 <sub>B</sub> The detection of a rising edge of Input Channel (2*i) generates a trigger event. INTF(2*i) becomes set
LDENO	10	rw	<b>Level Detection Enable 0</b> This bit determines if bit INTF(2i) is cleared automatically if an edge of the input Input Channel (2i) is detected, which has not been selected (rising edge with RENO = 0 or falling edge with FENO = 0). 0 <sub>B</sub> Bit INTF(2i) will not be cleared 1 <sub>B</sub> Bit INTF(2i) will be cleared
EIENO	11	rw	<b>External Input Enable 0</b> This bit enables the generation of a trigger event for request channel (2i) (e.g. for interrupt generation) when a selected edge is detected. 0 <sub>B</sub> The trigger event is disabled <b>1<sub>B</sub> The trigger event is enabled</b>

**System Control Units (SCU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>INP0</b>	14:12	rw	<p><b>Input Node Pointer</b></p> <p>This bit field determines the destination (output channel) for trigger event (2i) (if enabled by EIEN(2i)).</p> <p><b>000<sub>B</sub></b> An event from input ETL 2i triggers output OGU0 (signal TR(2i) 0)  <b>001<sub>B</sub></b> An event from input ETL 2i triggers output OGU1 (signal TR(2i) 1)  <b>010<sub>B</sub></b> An event from input ETL 2i triggers output OGU2 (signal TR(2i) 2)  <b>011<sub>B</sub></b> An event from input ETL 2i triggers output OGU3 (signal TR(2i) 3)  <b>100<sub>B</sub></b> An event from input ETL 2i triggers output OGU4 (signal TR(2i) 0)  <b>101<sub>B</sub></b> An event from input ETL 2i triggers output OGU5 (signal TR(2i) 0)  <b>110<sub>B</sub></b> An event from input ETL 2i triggers output OGU6 (signal TR(2i) 0)  <b>111<sub>B</sub></b> An event from input ETL 2i triggers output OGU7 (signal TR(2i) 0)</p>
<b>EXIS1</b>	22:20	rw	<p><b>External Input Selection 1</b></p> <p>This bit field determines which input line is selected for Input Channel (2i+1).</p> <p><b>000<sub>B</sub></b> Input (2i+1) A is selected  <b>001<sub>B</sub></b> Input (2i+1) B is selected  <b>010<sub>B</sub></b> Input (2i+1) C is selected  <b>011<sub>B</sub></b> Input (2i+1) D is selected  <b>100<sub>B</sub></b> Input (2i+1) E is selected  <b>101<sub>B</sub></b> Input (2i+1) F is selected  <b>110<sub>B</sub></b> Reserved  <b>111<sub>B</sub></b> Reserved</p>
<b>FEN1</b>	24	rw	<p><b>Falling Edge Enable 1</b></p> <p>This bit determines if the falling edge of Input Channel (2i+1) is used to set bit INTF(2i+1).</p> <p><b>0<sub>B</sub></b> The falling edge is not used  <b>1<sub>B</sub></b> The detection of a falling edge of Input Channel 1 generates a trigger event. INTF(2i+1) becomes set</p>
<b>REN1</b>	25	rw	<p><b>Rising Edge Enable 1</b></p> <p>This bit determines if the rising edge of Input Channel (2i+1) is used to set bit INTF(2i+1).</p> <p><b>0<sub>B</sub></b> The rising edge is not used  <b>1<sub>B</sub></b> The detection of a rising edge of Input Channel 1 generates a trigger event . INTF(2i+1) becomes set</p>
<b>LDEN1</b>	26	rw	<p><b>Level Detection Enable 1</b></p> <p>This bit determines if bit INTF(2i+1) is cleared automatically if an edge of the input Input Channel (2i+1) is detected, which has not been selected (rising edge with REN1 = 0 or falling edge with FEN1 = 0).</p> <p><b>0<sub>B</sub></b> Bit INTF(2i+1) will not be cleared  <b>1<sub>B</sub></b> Bit INTF1(2i+1) will be cleared</p>
<b>EIEN1</b>	27	rw	<p><b>External Input Enable 1</b></p> <p>This bit enables the generation of a trigger event for request channel (2i+1) (e.g. for interrupt generation) when a selected edge is detected.</p> <p><b>0<sub>B</sub></b> The trigger event is disabled  <b>1<sub>B</sub></b> The trigger event is enabled</p>

## System Control Units (SCU)

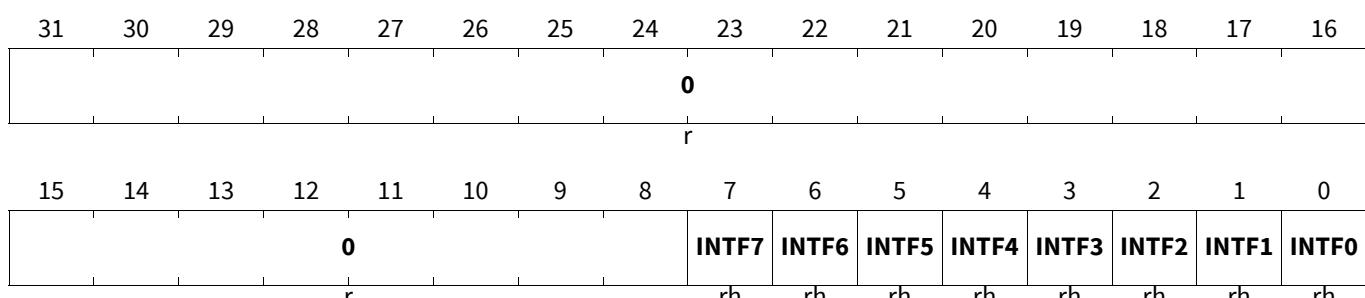
Field	Bits	Type	Description
<b>INP1</b>	30:28	rw	<b>Input Node Pointer</b> This bit field determines the destination (output channel) for trigger event (2i+1) (if enabled by EIEN(2i+1)). $000_B$ An event from input ETL 2i+1 triggers output OGU0 (signal TR(2i+1) 0) $001_B$ An event from input ETL 2i+1 triggers output OGU1 (signal TR(2i+1) 1) $010_B$ An event from input ETL 2i+1 triggers output OGU2 (signal TR(2i+1) 2) $011_B$ An event from input ETL 2i+1 triggers output OGU3 (signal TR(2i+1) 3) $100_B$ An event from input ETL 2i+1 triggers output OGU4 (signal TR(2i+1) 0) $101_B$ An event from input ETL 2i+1 triggers output OGU5 (signal TR(2i+1) 0) $110_B$ An event from input ETL 2i+1 triggers output OGU6 (signal TR(2i+1) 0) $111_B$ An event from input ETL 2i+1 triggers output OGU7 (signal TR(2i+1) 0)
<b>0</b>	3:0, 7, 19:15, 23, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

### External Input Flag Register

The External Input Flag Register EIFR contains all status flags for the external input channels. The bits in this register can be cleared by software by setting FMR.FCx, and set by setting FMR.FSx.

#### EIFR

#### External Input Flag Register (0220<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>INTFx (x=0-7)</b>	x	rh	<b>External Event Flag of Channel x</b> This bit monitors the status flag of the event trigger condition for the input channel x. This bit is automatically cleared when the selected condition (see RENx, FENx) is no longer met (if LDENx = 1) or remains set until it is cleared by software (if LDENx = 0).

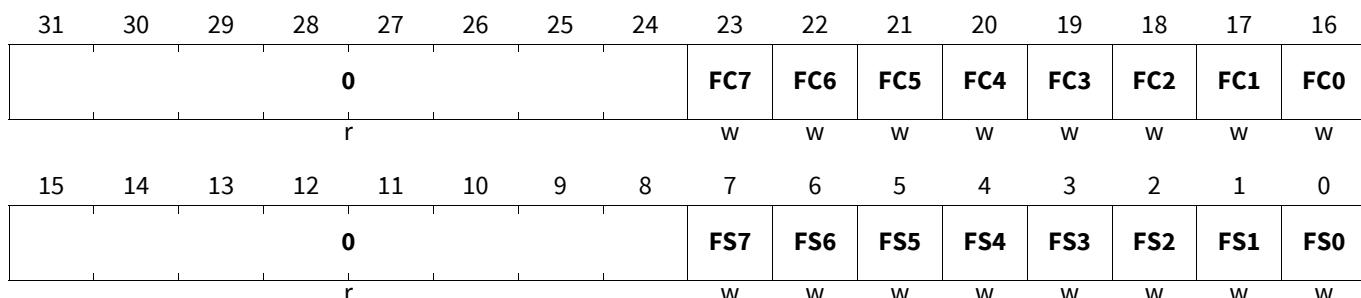
## System Control Units (SCU)

Field	Bits	Type	Description
0	31:8	r	<b>Reserved</b> Read as 0; should be written with 0.

### Flag Modification Register

FMR

Flag Modification Register

(0224<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
FSx (x=0-7)	x	w	<b>Set Flag INTFx for Channel x</b> Setting this bit will set the corresponding bit INTFx in register EIFR. Reading this bit always delivers a 0. If both FSx and FCx are set in the same access then the bit x in register EIFR is not modified. 0 <sub>B</sub> The bit x in register EIFR is not modified 1 <sub>B</sub> The bit x in register EIFR is set
FCx (x=0-7)	x+16	w	<b>Clear Flag INTFx for Channel x</b> Setting this bit will clear the corresponding bit INTFx in register EIFR. Reading this bit always delivers a 0. If both FSx and FCx are set in the same access then the bit x in register EIFR is not modified. 0 <sub>B</sub> The bit x in register EIFR is not modified 1 <sub>B</sub> The bit x in register EIFR is cleared
0	15:8, 31:24	r	<b>Reserved</b> Read as 0; should be written with 0.

### Pattern Detection Result Register

The Pattern Detection Result Register monitors the combinatorial output status of the pattern detection units.

## System Control Units (SCU)

### PDRR

#### Pattern Detection Result Register

(0228<sub>H</sub>)Application Reset Value: 0000 00FF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								PDR7	PDR6	PDR5	PDR4	PDR3	PDR2	PDR1	PDR0
r								rh							

Field	Bits	Type	Description
PDRy (y=0-7)	y	rh	<b>Pattern Detection Result of Channel y</b> This bit monitors the output status of the pattern detection for the output channel y.
0	31:8	r	<b>Reserved</b> Read as 0; should be written with 0.

### Flag Gating Register j

Each Interrupt Gating Control Registers IGCRj (j=0 to 3) contains bits to enable the pattern detection and to control the gating for two output channels.

### IGCRj (j=0-3)

#### Flag Gating Register j

(022C<sub>H</sub>+j\*4)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IGP1	GEEN1							IPEN1 7	IPEN1 6	IPEN1 5	IPEN1 4	IPEN1 3	IPEN1 2	IPEN1 1	IPEN1 0
rw	rw							rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IGP0	GEENO							IPENO 7	IPENO 6	IPENO 5	IPENO 4	IPENO 3	IPENO 2	IPENO 1	IPENO 0
rw	rw							rw							

Field	Bits	Type	Description
IPEN0x (x=0-7)	x	rw	<b>Flag Pattern Enable for Channel 0</b> Bit IPEN0x determines if the flag INTFx of channel x takes part in the pattern detection for the gating of the requests for the output signals GOUT(2*j) and IOUT(2*j). 0 <sub>B</sub> The bit INTFx does not take part in the pattern detection 1 <sub>B</sub> The bit INTFx is taken into consideration for the pattern detection

**System Control Units (SCU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>GEEN0</b>	13	rw	<p><b>Generate Event Enable 0</b></p> <p>Bit GEEN0 enables the generation of a trigger event for output channel (2*j) when the result of the pattern detection changes. When using this feature, a trigger (e.g. for an interrupt) is generated during the first clock cycle when a pattern is detected or when it is no longer detected.</p> <p>0<sub>B</sub> The trigger generation at a change of the pattern detection result is disabled</p> <p>1<sub>B</sub> The trigger generation at a change of the pattern detection result is enabled</p>
<b>IGP0</b>	15:14	rw	<p><b>Interrupt Gating Pattern 0</b></p> <p>In each register IGCRj, bit field IGP0 determines how the pattern detection influences the output lines GOUT(2j) and IOUT(2j).</p> <p>00<sub>B</sub> IOUT(2j) is inactive. The pattern is not considered.</p> <p>01<sub>B</sub> IOUT(2j) is activated in response to a trigger event. The pattern is not considered.</p> <p>10<sub>B</sub> The detected pattern is considered. IOUT(2j) is activated if a trigger event occurs while the pattern is present.</p> <p>11<sub>B</sub> The detected pattern is considered. IOUT(2j) is activated if a trigger event occurs while the pattern is not present.</p>
<b>IPEN1x (x=0-7)</b>	x+16	rw	<p><b>Interrupt Pattern Enable for Channel 1</b></p> <p>Bit IPEN(2j+1)x determines if the flag INTFx of channel (2j+1) takes part in the pattern detection for the gating of the requests for the output signals GOUT(2j+1) and IOUT(2j+1).</p> <p>0<sub>B</sub> The bit INTFx does not take part in the pattern detection</p> <p>1<sub>B</sub> The bit INTFx is taken into consideration for the pattern detection</p>
<b>GEEN1</b>	29	rw	<p><b>Generate Event Enable 1</b></p> <p>Bit GEEN1 enables the generation of a trigger event for output channel (2j+1) when the result of the pattern detection changes. When using this feature, a trigger (e.g. for an interrupt) is generated during the first clock cycle when a pattern is detected, or when it is no longer detected.</p> <p>0<sub>B</sub> The trigger generation at a change of the pattern detection result is disabled</p> <p>1<sub>B</sub> The trigger generation at a change of the pattern detection result is enabled</p>
<b>IGP1</b>	31:30	rw	<p><b>Interrupt Gating Pattern 1</b></p> <p>In each register IGCRj, bit field IGP1 determines how the pattern detection influences the output lines GOUT(2j+1) and IOUT(2j+1).</p> <p>00<sub>B</sub> IOUT(2j+1) is inactive. The pattern is not considered.</p> <p>01<sub>B</sub> IOUT(2j+1) is activated in response to a trigger event. The pattern is not considered.</p> <p>10<sub>B</sub> The detected pattern is considered. IOUT(2j+1) is activated if a trigger event occurs while the pattern is present.</p> <p>11<sub>B</sub> The detected pattern is considered. IOUT(2j+1) is activated if a trigger event occurs while the pattern is not present.</p>
<b>0</b>	12:8, 28:24	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

## **System Control Units (SCU)**

## 9.6 Emergency Stop (ES)

The Emergency Stop unit (ES) provides a fast reaction to an emergency event without the intervention of software. In response to the emergency event, selected output ports can be immediately placed into a defined state (for more information see the PORT chapter).

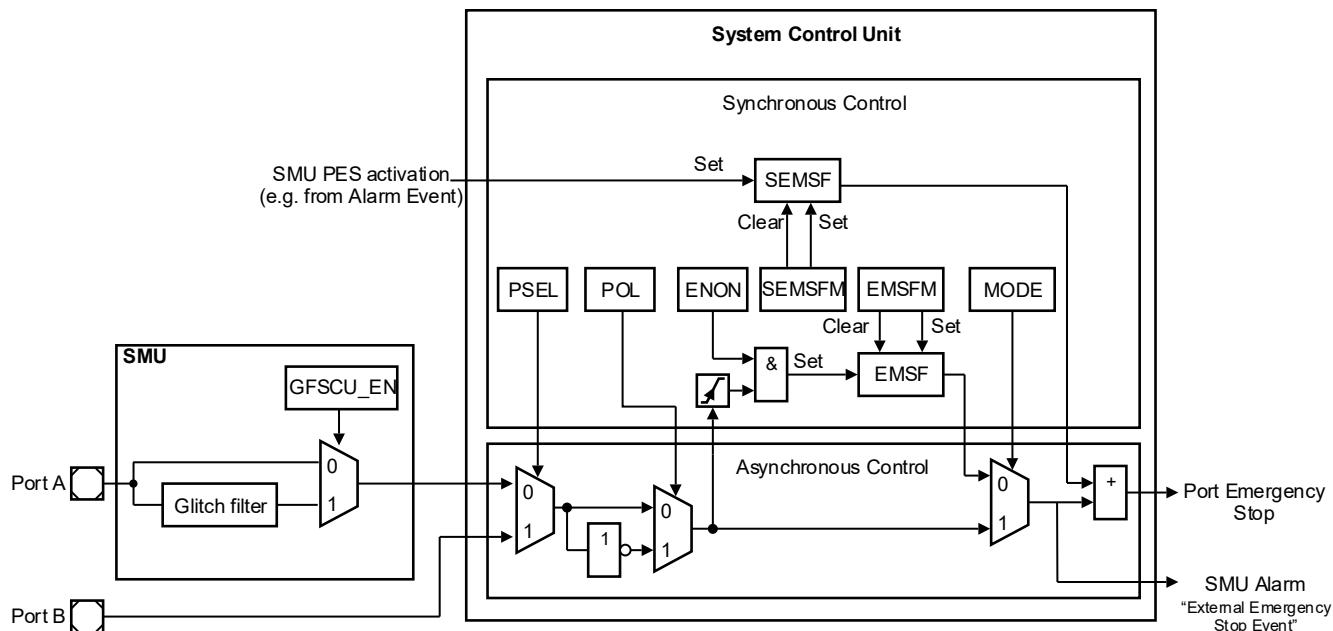
### **9.6.1 Feature List**

An Emergency Stop may be triggered by either of the following emergency events:

- A transition on the port which is configured as the Emergency Stop input
  - An SMU alarm event or SMU command which is enabled and configured to generate an Port Emergency Stop (PES). See SMU Chapter for details.

**Figure 80** shows a diagram of the emergency stop input logic. This logic is controlled by the Emergency Stop Register EMSR. There is also a possibility of using a Glitch filter, implemented in the SMU, for the Port A.

**Attention:** When a dynamic fault signaling protocol (FSP) is used, the Port A cannot be chosen as input trigger for the SCU



**Figure 80 Emergency Stop Control**

## 9.6.2 Delta to AURIX

The most significant changes between the TC2xx and TC3xx are:

- Glitch filter implemented in Port A Emergency Stop Input
  - EMSR register is now SE protected
  - The SW set and Clear fields are now in a separate register/address

### **9.6.3 Port Triggered Emergency Stop**

This can be configured to trigger on either transition edge of either one of two ports.

The two input port options are (A) P 33.8 and (B) P21.2.

## System Control Units (SCU)

The emergency stop control logic for the ports can basically operate in two modes:

- Synchronous Mode (default after reset):  
Emergency case is activated by hardware and released by software.
- Asynchronous Mode:  
Emergency case is activated and released by hardware.

In Synchronous Mode (selected by EMSR.MODE = 0), the port signal is sampled for a inactive-to-active level transition, and an emergency stop flag EMSR.EMSF is set if the transition is detected. The setting of EMSR.EMSF activates the emergency stop. A port triggered emergency state can only be terminated by clearing EMSR.EMSF via software (Write EMSSW.EMSFM with  $10_B$ ). The synchronous control logic is clocked by the system bus clock  $f_{SPB}$ . This results in a small delay between the port signal and emergency stop signal generation.

In Asynchronous Mode (selected by EMSR.MODE = 1), the occurrence of an active level at the port input immediately activates the emergency stop signal. Of course, a valid-to-invalid transition of the port input (emergency case is released) also immediately deactivates the emergency stop signal.

The EMSR.POL bit determines the active level of the input signal from the port. The EMSR.MODE bit selects Synchronous or Asynchronous Mode for emergency stop signal generation and the EMSR.PSEL bit selects which of the two ports is used as the emergency stop trigger.

### 9.6.4 SMU Event Triggered Emergency Stop

The Safety Alarm(s) which can trigger an Emergency Stop are configured and enabled within the Safety Management Unit (SMU). All SMU triggered Emergency Stop cases are in Synchronous Mode, regardless of the state of EMSR.MODE. The safety emergency stop flag EMSR.SEMSF is set when a configured and enabled SMU Safety Alarm occurs. The setting of EMSR.SEMSF activates the emergency stop. An SMU triggered emergency state can only be terminated by clearing the EMSR.SEMSF via software (Write EMSSW.SEMSF with  $10_B$ ). The synchronous control logic is clocked by the system bus clock  $f_{SPB}$ .

**System Control Units (SCU)****9.6.5 Emergency Stop Register****Emergency Stop Register****EMSR****Emergency Stop Register****(00FC<sub>H</sub>)****Application Reset Value: 0000 0001<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							0							SEMSF	EMSF
r														rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						0						PSEL	ENON	MODE	POL
r												rw	rw	rw	rw

Field	Bits	Type	Description
<b>POL</b>	0	rw	<b>Input Polarity</b> This bit determines the polarity of the configured Emergency Stop input. $0_B$ Input is active high $1_B$ Input is active low
<b>MODE</b>	1	rw	<b>Mode Selection</b> This bit determines the operating mode of the emergency stop signal. $0_B$ Synchronous Mode selected; emergency stop is derived from the state of flag EMSF $1_B$ Asynchronous Mode selected; emergency stop is directly derived from the state of the input signal
<b>ENON</b>	2	rw	<b>Enable ON</b> This bit enables the setting of flag EMSF by an inactive-to-active level transition of input signal. $0_B$ Setting of EMSF is disabled $1_B$ Setting of EMSF is enabled
<b>PSEL</b>	3	rw	<b>PORT Select</b> This bit selects which one of the two Emergency Stop port options is monitored. $0_B$ Port A is used as Emergency Stop input $1_B$ Port B is used as Emergency Stop input
<b>EMSF</b>	16	rh	<b>Emergency Stop Flag</b> This bit indicates that a synchronous mode port-triggered emergency stop condition has occurred. $0_B$ An emergency stop has not occurred $1_B$ An emergency stop has occurred and emergency stop state becomes active (if MODE = 0)

## System Control Units (SCU)

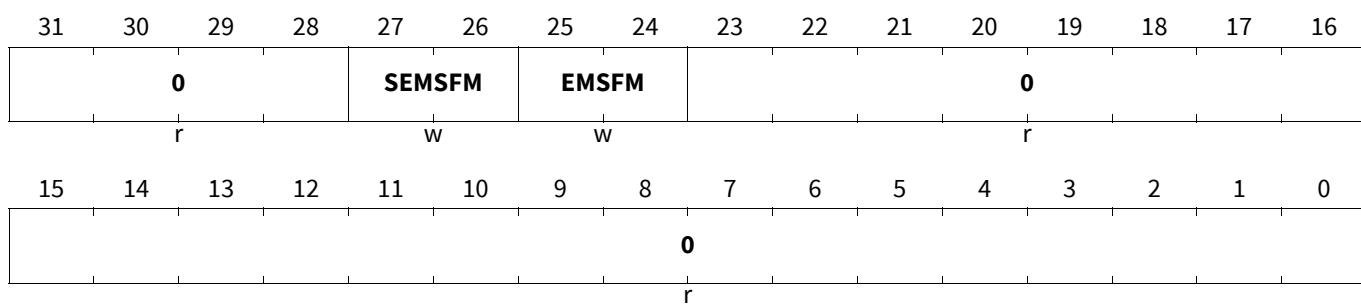
Field	Bits	Type	Description
<b>SEMSF</b>	17	rh	<b>SMU Emergency Stop Flag</b> This bit indicates that an SMU Safety Alarm triggered emergency stop condition has occurred. 0 <sub>B</sub> An emergency stop has not occurred 1 <sub>B</sub> An emergency stop has occurred and emergency stop state becomes active
<b>0</b>	15:4, 31:18	r	<b>Reserved</b> Read as 0; should be written with 0.

### Emergency Stop Software set and clear register

#### EMSSW

Emergency Stop Software set and clear register(0100<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>EMSF</b>	25:24	w	<b>Emergency Stop Flag Modification</b> This bit field sets or clears flag EMSF via software. EMSFM is always read as 00 <sub>B</sub> . 00 <sub>B</sub> EMSF remains unchanged 01 <sub>B</sub> EMSF becomes set 10 <sub>B</sub> EMSF becomes cleared 11 <sub>B</sub> EMSF remains unchanged
<b>SEMSFM</b>	27:26	w	<b>SMU Emergency Stop Flag Modification</b> This bit field sets or clears flag SEMSF via software. SEMSFM is always read as 00 <sub>B</sub> . 00 <sub>B</sub> SEMSF remains unchanged 01 <sub>B</sub> SEMSF becomes set 10 <sub>B</sub> SEMSF becomes cleared 11 <sub>B</sub> SEMSF remains unchanged
<b>0</b>	23:0, 31:28	r	<b>Reserved</b> Read as 0; should be written with 0.

---

**System Control Units (SCU)****9.7 Power Management Control Registers (PMC)**

Various control registers for Power Management are also part of the SCU.

For convenience these are described in the PMS subchapter “Power Management Control Registers (SCU)”.

**System Control Units (SCU)****9.8 Registers**

Note: Write access to LCLCONx registers are ignored without any further action when ST protection is active.

**Table 275 Register Overview - SCU (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
ID	Identification Register	0008 <sub>H</sub>	U,SV	BE	System Reset	<a href="#">64</a>
RSTSTAT	Reset Status Register	0050 <sub>H</sub>	U,SV	BE	See page <a href="#">11</a>	<a href="#">11</a>
RSTCON	Reset Configuration Register	0058 <sub>H</sub>	U,SV	SV,SE,P0	See page <a href="#">14</a>	<a href="#">14</a>
ARSTDIS	Application Reset Disable Register	005C <sub>H</sub>	U,SV	SV,E,P0	PowerOn Reset	<a href="#">16</a>
SWRSTCON	Software Reset Configuration Register	0060 <sub>H</sub>	U,SV	SV,E,P0	See page <a href="#">17</a>	<a href="#">17</a>
RSTCON2	Additional Reset Control Register	0064 <sub>H</sub>	U,SV	SV,E,P0	See page <a href="#">18</a>	<a href="#">18</a>
RSTCON3	Reset Configuration Register 3	0068 <sub>H</sub>	U,SV	SV,E,P0	See page <a href="#">19</a>	<a href="#">19</a>
ESRCFGx	ESRx Input Configuration Register	0070 <sub>H</sub> +x *4	U,SV	SV,E,P0	System Reset	<a href="#">22</a>
ESROCFG	ESR Output Configuration Register	0078 <sub>H</sub>	U,SV	SV,E,P0	System Reset	<a href="#">22</a>
SYSCON	System Control Register	007C <sub>H</sub>	U,SV	U,SV,P0	System Reset	<a href="#">60</a>
PDR	ESR Pad Driver Mode Register	009C <sub>H</sub>	U,SV	SV,E,P0	System Reset	<a href="#">30</a>
IOCR	Input/Output Control Register	00A0 <sub>H</sub>	U,SV	U,SV,P0	System Reset	<a href="#">24</a>
OUT	ESR Output Register	00A4 <sub>H</sub>	U,SV	U,SV,P0	System Reset	<a href="#">27</a>
OMR	ESR Output Modification Register	00A8 <sub>H</sub>	U,SV	U,SV,P0	System Reset	<a href="#">28</a>
IN	ESR Input Register	00AC <sub>H</sub>	U,SV	BE	System Reset	<a href="#">29</a>
STSTAT	Start-up Status Register	00C0 <sub>H</sub>	U,SV	BE	PowerOn Reset	<a href="#">32</a>
STCON	Start-up Configuration Register	00C4 <sub>H</sub>	U,SV	ST,P0	Application Reset	<a href="#">34</a>
EMSR	Emergency Stop Register	00FC <sub>H</sub>	U,SV	SV,SE,P0	Application Reset	<a href="#">123</a>
EMSSW	Emergency Stop Software set and clear register	0100 <sub>H</sub>	U,SV	U,SV,P0	Application Reset	<a href="#">124</a>
TRAPDIS1	Trap Disable Register 1	0120 <sub>H</sub>	U,SV	SV,E,P0	Application Reset	<a href="#">41</a>
TRAPSTAT	Trap Status Register	0124 <sub>H</sub>	U,SV	BE	System Reset	<a href="#">37</a>
TRAPSET	Trap Set Register	0128 <sub>H</sub>	U,SV	SV,E,P0	System Reset	<a href="#">38</a>

**System Control Units (SCU)****Table 275 Register Overview - SCU (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
TRAPCLR	Trap Clear Register	012C <sub>H</sub>	U,SV	U,SV,P0	System Reset	<a href="#">39</a>
TRAPDIS0	Trap Disable Register 0	0130 <sub>H</sub>	U,SV	SV,E,P0	Application Reset	<a href="#">39</a>
LCLCON0	LCL CPU0 and CPU2 Control Register	0134 <sub>H</sub>	U,SV	SV,SE,ST,P0	See page <a href="#">43</a>	<a href="#">43</a>
LCLCON1	LCL CPU1 and CPU3 Control Register	0138 <sub>H</sub>	U,SV	SV,SE,ST,P0	See page <a href="#">45</a>	<a href="#">45</a>
LCLTEST	LCL Test Register	013C <sub>H</sub>	U,SV	U,SV,P0	System Reset	<a href="#">46</a>
CHIPID	Chip Identification Register	0140 <sub>H</sub>	U,SV	ST,P0	See page <a href="#">61</a>	<a href="#">61</a>
MANID	Manufacturer Identification Register	0144 <sub>H</sub>	U,SV	BE	System Reset	<a href="#">64</a>
SWAPCTRL	Address Map Control Register	014C <sub>H</sub>	U,SV	ST,P0	System Reset	<a href="#">70</a>
LBISTCTRL0	Logic BIST Control 0 Register	0164 <sub>H</sub>	U,SV	SV,SE,P0	See page <a href="#">51</a>	<a href="#">51</a>
LBISTCTRL1	Logic BIST Control 1 Register	0168 <sub>H</sub>	U,SV	SV,SE,P0	See page <a href="#">53</a>	<a href="#">53</a>
LBISTCTRL2	Logic BIST Control 2 Register	016C <sub>H</sub>	U,SV	SV,SE,P0	See page <a href="#">54</a>	<a href="#">54</a>
LBISTCTRL3	Logic BIST Control 3 Register	0170 <sub>H</sub>	U,SV	BE	See page <a href="#">55</a>	<a href="#">55</a>
STMEM1	Start-up Memory Register 1	0184 <sub>H</sub>	U,SV	ST,P0	PowerOn Reset	<a href="#">65</a>
STMEM2	Start-up Memory Register 2	0188 <sub>H</sub>	U,SV	ST,P0	System Reset	<a href="#">65</a>
PDISC	Pad Disable Control Register	018C <sub>H</sub>	U,SV	SV,E,P0	System Reset	<a href="#">29</a>
STMEM3	Start-up Memory Register 3	01C0 <sub>H</sub>	U,SV	ST,P0	Application Reset	<a href="#">65</a>
STMEM4	Start-up Memory Register 4	01C4 <sub>H</sub>	U,SV	ST,P0	Cold PowerOn Reset	<a href="#">66</a>
STMEM5	Start-up Memory Register 5	01C8 <sub>H</sub>	U,SV	ST,P0	PowerOn Reset	<a href="#">66</a>
STMEM6	Start-up Memory Register 6	01CC <sub>H</sub>	U,SV	ST,P0	System Reset	<a href="#">67</a>
OVCENABLE	Overlay Enable Register	01E0 <sub>H</sub>	U,SV	SV,SE,P0	Application Reset	<a href="#">56</a>
OVCCON	Overlay Control Register	01E4 <sub>H</sub>	U,SV	SV,P0	Application Reset	<a href="#">57</a>
EIFILT	External Input Filter Register	020C <sub>H</sub>	U,SV	SE,SV,P0	Application Reset	<a href="#">113</a>
EICRi	External Input Channel Register i	0210 <sub>H</sub> +i*4	U,SV	SE,SV,P0	Application Reset	<a href="#">114</a>

**System Control Units (SCU)****Table 275 Register Overview - SCU (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
EIFR	External Input Flag Register	0220 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">117</a>
FMR	Flag Modification Register	0224 <sub>H</sub>	U,SV	U,SV,P0	Application Reset	<a href="#">118</a>
PDRR	Pattern Detection Result Register	0228 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">118</a>
IGCRj	Flag Gating Register j	022C <sub>H</sub> +j*4	U,SV	SE,SV,P0	Application Reset	<a href="#">119</a>
WDTCPUyCON0	CPUy WDT Control Register 0	024C <sub>H</sub> +y*12	U,SV	U,SV,32,CP Uy (y=CPU number)	See page <a href="#">84</a>	<a href="#">84</a>
WDTCPUyCON1	CPUy WDT Control Register 1	0250 <sub>H</sub> +y*12	U,SV	SV,CEy,P0	See page <a href="#">88</a>	<a href="#">88</a>
WDTCPUySR	CPUy WDT Status Register	0254 <sub>H</sub> +y*12	U,SV	BE	See page <a href="#">93</a>	<a href="#">93</a>
EICON0	ENDINIT Global Control Register 0	029C <sub>H</sub>	U,SV	U,SV,32,P0	Application Reset	<a href="#">95</a>
EICON1	ENDINIT Global Control Register 1	02A0 <sub>H</sub>	U,SV	SV,E,P0	Application Reset	<a href="#">96</a>
EISR	ENDINIT Timeout Counter Status Register	02A4 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">97</a>
WDTSCON0	Safety WDT Control Register 0	02A8 <sub>H</sub>	U,SV	U,SV,32,P1	Application Reset	<a href="#">83</a>
WDTSCON1	Safety WDT Control Register 1	02AC <sub>H</sub>	U,SV	SV,SE,P1	Application Reset	<a href="#">86</a>
WDTSSR	Safety WDT Status Register	02B0 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">91</a>
SEICON0	Safety ENDINIT Control Register 0	02B4 <sub>H</sub>	U,SV	U,SV,32,P1	Application Reset	<a href="#">98</a>
SEICON1	Safety ENDINIT Control Register 1	02B8 <sub>H</sub>	U,SV	SV,SE,P1	Application Reset	<a href="#">100</a>
SEISR	Safety ENDINIT Timeout Status Register	02BC <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">101</a>
ACCEN11	Access Enable Register 11	03F0 <sub>H</sub>	U,SV	SV,SE	Application Reset	<a href="#">69</a>
ACCEN10	Access Enable Register 10	03F4 <sub>H</sub>	U,SV	SV,SE	Application Reset	<a href="#">68</a>
ACCEN01	Access Enable Register 01	03F8 <sub>H</sub>	U,SV	SV,SE	Application Reset	<a href="#">68</a>
ACCEN00	Access Enable Register 00	03FC <sub>H</sub>	U,SV	SV,SE	Application Reset	<a href="#">68</a>

## System Control Units (SCU)

### 9.8.1 Safety Flip-Flops

Safety flip-flops are special flip-flops that implement a hardware mechanism capable to detect single event effects, that may lead to single event upsets (bit flip). The configuration and control registers that are implemented with safety flip-flops are:

- **EMSR[16]**
- **EMSR[17]**
- **EMSR[3:0]**
- **LBISTCTRL0**
- **LBISTCTRL1**
- **LBISTCTRL2**
- **LBISTCTRL3**
- **CHIPID**
- **TRAPSTAT**
- **TRAPPDIS0**
- **TRAPPDIS1**
- **ESRCFGx (x=0-1)[8:7]**
- **ESROCFG[0]**
- **IOCR**
- **WDTCPUyCON1 (y=0-5)**
- **WDTSCON1**
- **EICON1**
- **SEICON1**
- **WDTCPUyCON0 (y=0-5)[0]**
- **WDTSCON0[0]**
- **EICON0[1]**
- **SEICON0[1]**

### 9.9 IO Interfaces

The table below lists all the interfaces of the SCU to other modules in the device.

**Table 276 List of SCU Interface Signals**

Interface Signals	I/O	Description
ERU_INT(3:0)	out	<b>SCU ERU Service Request x</b>
E_REQ0(5:0)	in	<b>ERU Channel 0 input X; x=0-5, where 0 is input A and 5 is input F.</b>
E_REQ1(5:0)	in	<b>ERU Channel 1 input X; x=0-5, where 0 is input A and 5 is input F.</b>
E_REQ2(5:0)	in	<b>ERU Channel 2 input X; x=0-5, where 0 is input A and 5 is input F.</b>
E_REQ3(5:0)	in	<b>ERU Channel 3 input X; x=0-5, where 0 is input A and 5 is input F.</b>

## System Control Units (SCU)

**Table 276 List of SCU Interface Signals (cont'd)**

Interface Signals	I/O	Description
E_REQ4(5:0)	in	<b>ERU Channel 4 input X; x=0-5, where 0 is input A and 5 is input F.</b>
E_REQ5(5:0)	in	<b>ERU Channel 5 input X; x=0-5, where 0 is input A and 5 is input F.</b>
E_REQ6(5:0)	in	<b>ERU Channel 6 input X; x=0-5, where 0 is input A and 5 is input F.</b>
E_REQ7(5:0)	in	<b>ERU Channel 7 input X; x=0-5, where 0 is input A and 5 is input F.</b>
E_IOUT(7:0)	out	<b>ERU IOUTn output (MSB is IOUT7 and LSB is IOUT0)</b>
E_PDOT(7:0)	out	<b>ERU PDOUTn output (MSB is PDOUT7 and LSB is PDOUT0)</b>
EMGSTOP_PORT_A	in	<b>Emergency stop Port Pin A input request</b>
EMGSTOP_PORT_B	in	<b>Emergency stop Port Pin B input request</b>
ESR0_PORT_IN	in	<b>ESR0 Port Pin input - can be used to trigger a reset or an NMI</b>
ESR1_PORT_IN	in	<b>ESR1 Port Pin input - can be used to trigger a reset or an NMI</b>
SMU_EMGSTP_REQ	in	<b>Emergency stop request from SMU</b>
SMU_TRAP_REQ	in	<b>TRAP request from the SMU</b>
CBS_ENDINIT_DIS	in	<b>Watchdog ENDINIT disable from Cerberus</b>
CBS_WDT_SUSP	in	<b>Watchdog suspend from Cerberus</b>
RST_REQ_STM(5:0)	in	<b>Reset request from STMn (MSB is STM5 and LSB is STM0)</b>
TRAP_CPU(5:0)	out	<b>TRAP output to CPUn (MSB is CPU5 and LSB is CPU0)</b>

## 9.10 Revision History

### 9.10.1 SCU Complete Revision History

This is the complete revision history of the SCU. It contains all the relevant functional modifications for all devices. For a specific device revision history, one can address the Appendix revision history.

**Table 277 Revision History**

Reference	Change to Previous Version	Comment
V2.1.21	Revision History entries up to V2.1.20 removed.	
<a href="#">Page 126</a>	Added the Safety Flip-Flop Section	
<a href="#">Page 126</a>	LBIST reset termination textual description was updated (still missing the part of the register reset overview)	
<a href="#">Page 126</a>	Removed "Attention" note referring to TESTMODE pin.	
<a href="#">Page 18</a>	Added description "For products with no MCDS/miniMCDS/MCDSlight, the bit will always read '1'." to RSTCON2.MCDSS and RSTCON2.CSSx bitfields.	

## System Control Units (SCU)

**Table 277 Revision History (cont'd)**

Reference	Change to Previous Version	Comment
<a href="#">Page 31</a>	Note referring to TESTMODE pin has been removed from documentation.	
<b>V2.1.22</b>		
	Revision History entries up to V2.1.22 removed.	
<a href="#">Page 11</a>	Cold PORST reset value changed from 1xx10000 to 0xx10000 in RSTSTAT register.	
<a href="#">Page 11</a>	Additional cold_power_on_reset value “LVD Reset” added to RSTSTAT register.	
<a href="#">Page 61</a>	SCU_CHIPID register, CHID bitfield: Entered description of CHID constant value $E_H$ (14) for TC3Exx: “ $E_H$ SAx-TC3Exx”.	
<a href="#">Page 61</a>	SCU_CHIPID register, CHPK bitfield: Corrected description for constant value “2” to TQFP-80.	
<a href="#">Page 51</a>	LBISTCTRL0: System Reset value set to “Internal”. Added note to CFS Value in reset table: “Value installed after System and Power-On Reset.”	
<a href="#">Page 43</a>	LCLCON0 and LCLCON1: Cold PORST reset table values updated/corrected.	
<a href="#">Page 74</a>	Several typos of “ENDINIT” corrected.	
<a href="#">Page 43</a>	LCLCON0 and LCLCON1: Cold PORST reset table values updated/corrected.	
<b>V2.1.23</b>		
	Revision History entries up to V2.1.21 removed.	
	TC38EVOX product name changed to TC3Ex.	
	Typo “Value” corrected in Revision History V2.1.22.	
<a href="#">Page 70</a>	Corrected “ENDINIT Timeout Counter register (EICON0)” to “ENDINIT Global Control Registers (EICONx)” in section “Access to Endinit-Protected Registers without using WDT”.	
<a href="#">Page 129</a>	Improved the generic description of SCU_E_REQ x_request pins for all ERU channel X input descriptions, e.g.: ‘ERU channel 2 input ; x=0-5, where 0 is input A and 5 is input F.’.	
<a href="#">Page 43</a>	Typo “Mhz” corrected to “MHz” in SRU Functional Description.	
<a href="#">Page 43</a>	SCU_CHIPID register, CHID bitfield updated (clean-up, SAx-TC3Exx new) in section “Identification Registers”.	
<a href="#">Page 43</a>	Bitfield “LBISTREQRED” in register LBISTCTRL0 changed from “w” to “rw”.	
<a href="#">Page 43</a>	Masked LBISTCTRL0 reset value to display “000-----” instead of “0000-0000” in “Reset Values of LBISTCTRL0” table. Added note “The correct reset value of the LBISTCTRL0 register has to be looked up in the product-specific appendix document.”.	
<a href="#">Page 43</a>	Added information regarding stickyness of LBISTCTRL0.LBISTDONE bit to System and Application resets in section “LBIST Support”.	
<b>V2.1.24</b>		
<a href="#">Page 126</a>	Added note regarding register LCLCONx in Registers chapter.	

## System Control Units (SCU)

**Table 277 Revision History (cont'd)**

Reference	Change to Previous Version	Comment
<a href="#">Page 43</a>	Added note regarding STCON.STP bit field in chapter SRU.	
<a href="#">Page 43</a>	Updated reset values of LBISTCTRL0 register in chapter SRU.	
<a href="#">Page 35</a>	Added an inversion to CPU Trap Generation figure.	
<a href="#">Page 3</a>	Added overline to STSTAT.TRST bit field description instead of 'TRSTL'.	
<a href="#">Page 43</a>	Updated register EMDS to EMSSW twice in chapter SRU section "Emergency Stop".	

### V2.1.25

<a href="#">Page 3</a>	Updated figure "ESR Operation" in section "External Service Requests (ESRx)".	
<a href="#">Page 43</a>	Changed CHIPID.CHPK[11:8] value A to "BGA216" and CHIPID.CHID[15:12] value A to "SAx-TC3Axx".	
<a href="#">Page 102</a>	Updated description in chapter ERU section "Introduction" and added table to Chapter Pattern Detection.	

## Clocking System

# 10 Clocking System

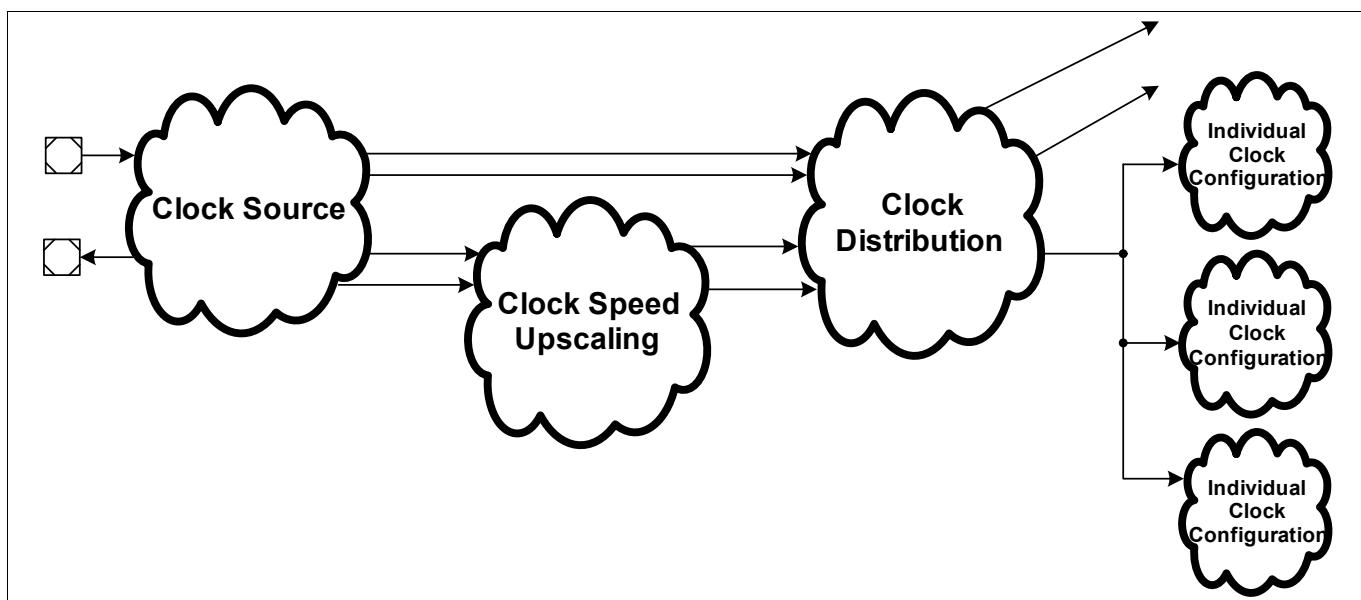
This section describes the clock system, its configuration, and the principles upon which it is based.

## 10.1 Overview

The clock system itself is built up as a chain composed from different building blocks which allow different certain function parts for this complete chain.

The building blocks are:

- Basic clock generation (Clock Source)
- Clock speed up-scaling (PLLs)
- Clock distribution (CCU)
- Individual clock configuration (Peripherals)



**Figure 81 Clock Tree**

Aside from the pure clock generation options, there are several support functions which have been integrated in order to support easy and convenient controls.

In the following subsections the clock tree is explained from left to right. Using this flow for the initial configuration is also recommended. Expert users can of course execute the configuration in an individual order.

**Note:** *If a running system needs to be re-configured not all parts of the clock tree need to be configured. Only these parts that are required can be updated.*

## Clocking System

### 10.2 Clocking System Registers Overview

The following table lists all SCU registers which are specific to the clocking system. They are prefixed with *SCU\_* in the register files.

**Table 278 Register Overview - CCU (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
OSCCON	OSC Control Register	0010 <sub>H</sub>	U,SV	SV,SE,P0	See page 6	6
SYSPLLSTAT	System PLL Status Register	0014 <sub>H</sub>	U,SV	BE	See page 14	14
SYSPLLCNO	System PLL Configuration 0 Register	0018 <sub>H</sub>	U,SV	SV,SE,P0	System Reset	15
SYSPLLCN1	System PLL Configuration 1 Register	001C <sub>H</sub>	U,SV	SV,SE,P0	System Reset	16
SYSPLLCN2	System PLL Configuration 2 Register	0020 <sub>H</sub>	U,SV	SV,SE,P0	System Reset	17
PERPLLSTAT	Peripheral PLL Status Register	0024 <sub>H</sub>	U,SV	BE	System Reset	20
PERPLLCNO	Peripheral PLL Configuration 0 Register	0028 <sub>H</sub>	U,SV	SV,SE,P0	System Reset	21
PERPLLCN1	Peripheral PLL Configuration 1 Register	002C <sub>H</sub>	U,SV	SV,SE,P0	System Reset	22
CCUCONO	CCU Clock Control Register 0	0030 <sub>H</sub>	U,SV	SV,SE,P0	See page 28	28
CCUCON1	CCU Clock Control Register 1	0034 <sub>H</sub>	U,SV	SV,SE,P0	System Reset	32
FDR	Fractional Divider Register	0038 <sub>H</sub>	U,SV	SV,SE,P0	System Reset	49
EXTCON	External Clock Control Register	003C <sub>H</sub>	U,SV	SV,SE,P0	System Reset	47
CCUCON2	CCU Clock Control Register 2	0040 <sub>H</sub>	U,SV	SV,SE,P0	System Reset	36
CCUCON3	CCU Clock Control Register 3	0044 <sub>H</sub>	U,SV	SV,SE,P0	System Reset	53
CCUCON4	CCU Clock Control Register 4	0048 <sub>H</sub>	U,SV	SV,SE,P0	System Reset	55
CCUCON5	CCU Clock Control Register 5	004C <sub>H</sub>	U,SV	SV,SE,P0	System Reset	39
CCUCON6	CCU Clock Control Register 6	0080 <sub>H</sub>	U,SV	SV,SE,P0	System Reset	41
CCUCON7	CCU Clock Control Register 7	0084 <sub>H</sub>	U,SV	SV,SE,P0	System Reset	42
CCUCON8	CCU Clock Control Register 8	0088 <sub>H</sub>	U,SV	SV,SE,P0	System Reset	42
CCUCON9	CCU Clock Control Register 9	008C <sub>H</sub>	U,SV	SV,SE,P0	System Reset	43

## Clocking System

**Table 278 Register Overview - CCU (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
CCUCON10	CCU Clock Control Register 10	0090 <sub>H</sub>	U,SV	SV,SE,P0	System Reset	<b>43</b>
CCUCON11	CCU Clock Control Register 11	0094 <sub>H</sub>	U,SV	SV,SE,P0	System Reset	<b>44</b>

### 10.2.1 Safety Flip-Flops

Safety flip-flops are special flip-flops that implement a hardware mechanism capable to detect single event effects that may lead to single event upsets (bit flip). The configuration and control registers that are implemented with safety flip-flops are:

- [OSCCON](#)
- [SYSPLLCON0](#)
- [SYSPLLCON1](#)
- [SYSPLLCON2](#)
- [PERPLLCON0](#)
- [PERPLLCON1](#)
- [CCUCON0](#)
- [CCUCON1](#)
- [CCUCON2](#)
- [CCUCON3](#)
- [CCUCON4](#)
- [CCUCON5](#)
- [CCUCON6](#)
- [CCUCON7](#)
- [CCUCON8](#)
- [CCUCON9](#)
- [CCUCON10](#)
- [CCUCON11](#)

### 10.3 Clock Sources

There are several clock sources available which either source the complete device, a major or minor part, or only dedicated modules. This depends on the sources and the configuration.

Please note that several clock sources can be used in parallel inside the system, but the main function of each peripheral is only related to one source at any time.

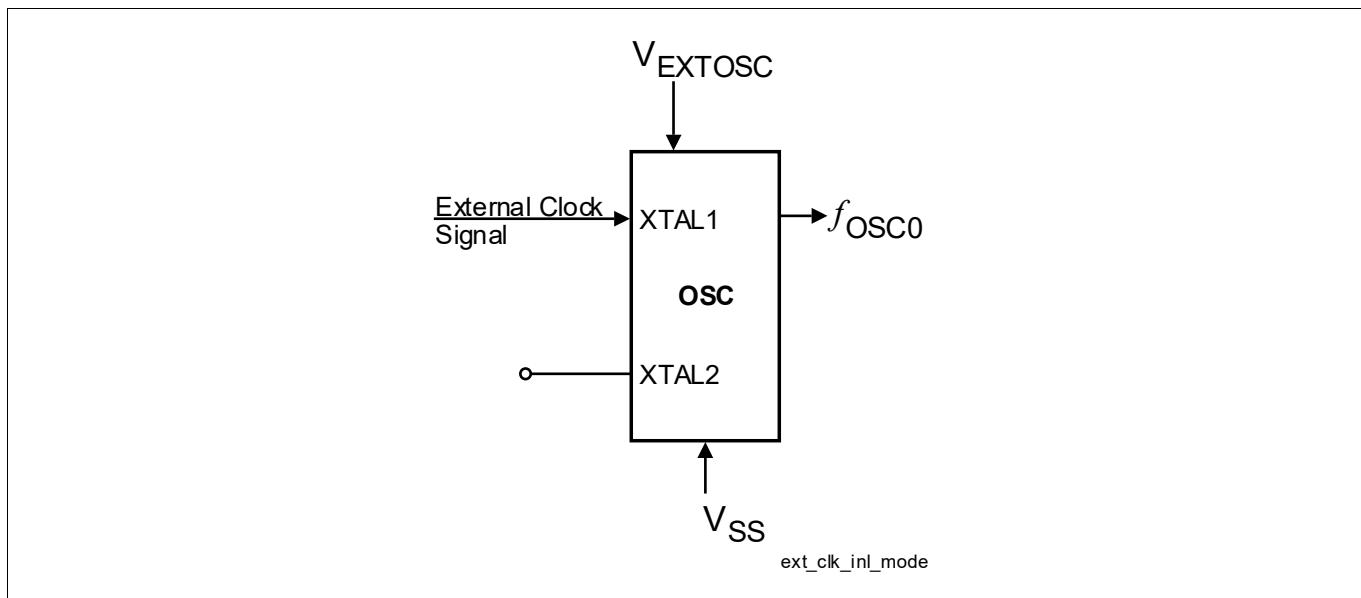
#### 10.3.1 Oscillator Circuit (OSC)

The oscillator circuit, a Pierce oscillator, is designed to work with both an external crystal / ceramic resonator or an external stable clock source. The circuit consists of an inverting amplifier with XTAL1 as input, and XTAL2 as output with an integrated feedback resistor.

## Clocking System

### 10.3.1.1 External Input Clock Mode

When using an external clock signal it must be connected to XTAL1. XTAL2 is left open (unconnected).

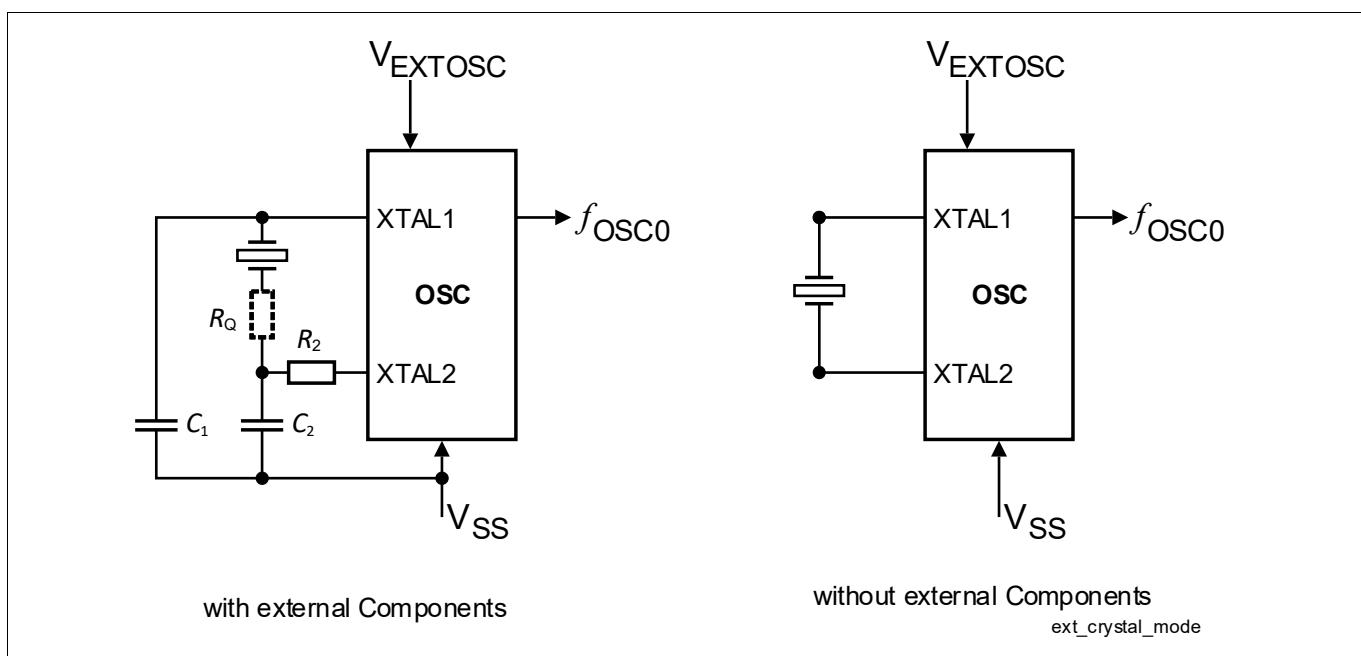


**Figure 82 AURIX™ TC3xx Platform Direct Clock Input**

When supplying the clock signal directly, not using an external crystal / ceramic resonator and bypassing the oscillator, the input frequency needs to be equal or greater than the PLL's DCO input frequency (the value is listed in the Data Sheet) if used in normal Mode.

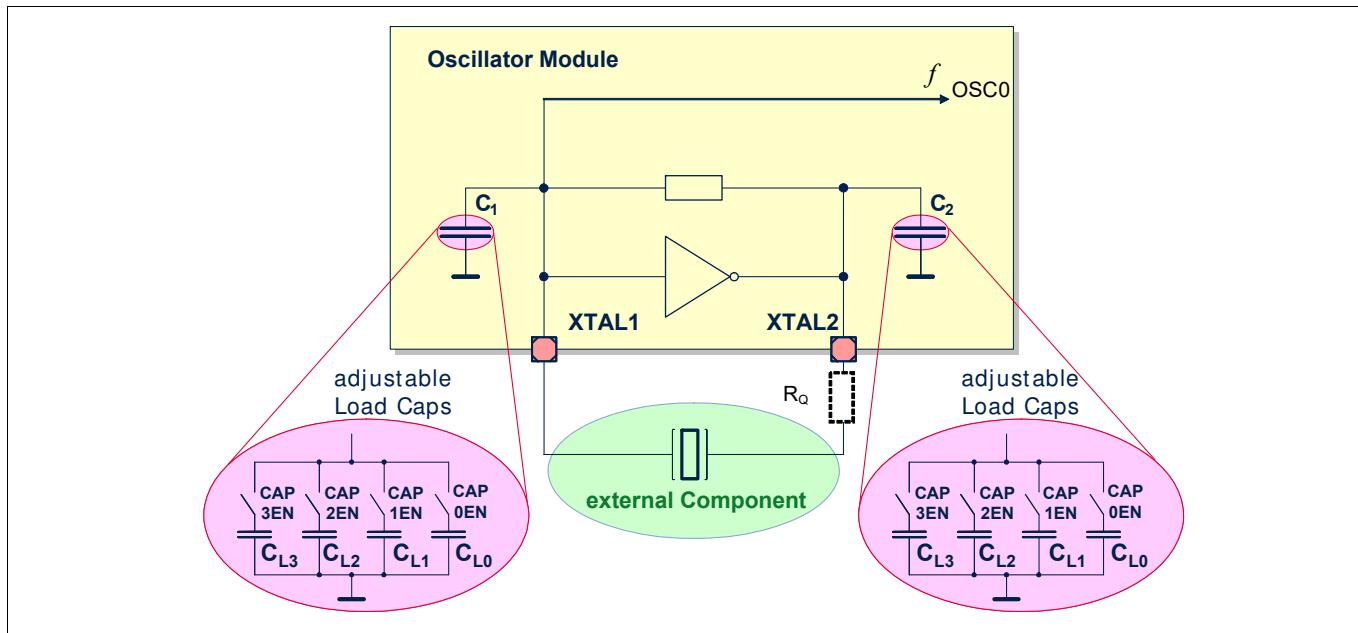
### 10.3.1.2 External Crystal / Ceramic Resonator Mode

**Figure 83** shows the recommended external circuits for both operating modes: External Crystal / Ceramic Resonator Mode with and without external components.



**Figure 83 External Circuitry for Crystal / Ceramic Resonator operation**

## Clocking System



**Figure 84 Circuitry for Crystal / Ceramic Resonator operation with internal caps**

When using an external crystal / ceramic resonator, its frequency can be within the allowed range (the values are listed in the Data Sheet). An external oscillator load circuitry can be used, connected to both pins, XTAL1 and XTAL2. Also needed are two load capacitors  $C_1$  and  $C_2$  (see [Figure 83](#)) or the internal loads can be used (see [Figure 84](#)). Also, depending on the crystal / ceramic resonator type, a series resistor  $R_2$  is needed to limit the current. A test resistor  $R_Q$  may be temporarily inserted to measure the oscillation allowance (negative resistance) of the oscillator circuitry.  $R_Q$  values are typically specified by the crystal / ceramic resonator vendor. The  $C_1$  and  $C_2$  values shown in the Data Sheet can be used as starting points for the negative resistance evaluation and for non-production systems. The exact values and related operating range are dependent on the crystal / ceramic resonator frequency, and have to be determined and optimized together with the crystal / ceramic resonator vendor using the negative resistance method. Oscillation measurement with the final target system is strongly recommended to verify the input amplitude at XTAL1 and to determine the actual oscillation allowance (margin negative resistance) for the oscillator-crystal / ceramic resonator system.

The oscillator can also be used in combination with a ceramic resonator. The final circuitry must be also verified by the resonator vendor.

## Clocking System

### 10.3.1.3 Oscillator Circuit Control Register

#### OSC Control Register

##### OSCCON

##### OSC Control Register

(0010<sub>H</sub>)Reset Value: [Table 279](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				<b>CAP3E</b> N	<b>CAP2E</b> N	<b>CAP1E</b> N	<b>CAP0E</b> N	<b>APRE</b> N		<b>0</b>					
r				rw	rw	rw	rw	rw	r						rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>0</b>		<b>AMPCTL</b>		<b>HYSCTL</b>	<b>HYSEN</b>	<b>PLLHV</b>	<b>SHBY</b>		<b>MODE</b>		<b>GAINSEL</b>	<b>OSCR</b> <b>ES</b>	<b>PLLLV</b>		<b>0</b>
r		rw		rw	rw	rh	rw	rw	rw		rw	w	rh	r	

Field	Bits	Type	Description
<b>PLLLV</b>	1	rh	<p><b>Oscillator for PLL Valid Low Status Bit</b>  This bit indicates if the frequency output <math>f_{osc}</math> of the oscillator is above the lower threshold frequency <math>f_{LV}</math>, i.e. usable for the DCO part of the PLL. This is checked by the Oscillator Watchdog of the PLL using the backup clock <math>f_{BACK}</math>.  By using the crystal's nominal frequency (<math>f_{oscnom}</math>), the lower threshold calculates as follows:</p> <p>Note: <math>f_{LV} = f_{oscnom} * 0,96 - 0,31</math></p> <p><math>0_B</math> The OSC frequency is not usable. Frequency <math>f_{osc}</math> is too low.  <math>1_B</math> The OSC frequency is usable</p>
<b>OSCRS</b>	2	w	<p><b>Oscillator Watchdog Reset</b>  Note: Always read as zero.</p> <p><math>0_B</math> The Oscillator Watchdog of the PLL is not cleared and remains active  <math>1_B</math> The Oscillator Watchdog of the PLL is cleared and restarted</p>

## Clocking System

Field	Bits	Type	Description
<b>GAINSEL</b>	4:3	rw	<p><b>Oscillator Gain Selection</b>            In Normal Mode this value should not be changed from the reset value <math>11_B</math>.</p> <p><b>Note:</b> When using <math>V_{ext}=3.3V</math>, the LVDS bias distributor has to be adjusted to 3.3V supply via <math>P21\_LPCR2.PS = 0</math> otherwise the oscillator gain can be too low for a reliable oscillator startup at cold temperature. In case of using <math>V_{ext}=5V</math>, the LVDS bias distributor setting stays at the reset value <math>P21\_LPCR2.PS = 1</math>.</p> <ul style="list-style-type: none"> <li><math>00_B</math> Low gain 1; Reserved for future use</li> <li><math>01_B</math> Low gain 2; Reserved for future use</li> <li><math>10_B</math> Low gain 3; Reserved for future use</li> <li><math>11_B</math> Maximum gain configuration</li> </ul>
<b>MODE</b>	6:5	rw	<p><b>Oscillator Mode</b>            This bit field defines which mode can be used and if the oscillator entered the Power-Saving Mode or not.</p> <ul style="list-style-type: none"> <li><math>00_B</math> External Crystal / Ceramic Resonator Mode. The oscillator Power-Saving Mode is not entered.</li> <li><math>01_B</math> OSC is disabled. The oscillator Power-Saving Mode is not entered.</li> <li><math>10_B</math> External Input Clock Mode and the oscillator Power-Saving Mode is entered</li> <li><math>11_B</math> OSC is disabled. The oscillator Power-Saving Mode is entered.</li> </ul>
<b>SHBY</b>	7	rw	<p><b>Shaper Bypass</b>  <math>0_B</math> The shaper is not bypassed (default)  <math>1_B</math> The shaper is bypassed (reserved)</p>
<b>PLLHV</b>	8	rh	<p><b>Oscillator for PLL Valid High Status Bit</b>            This bit indicates if the frequency output <math>f_{osc}</math> of the oscillator is below the upper threshold frequency <math>f_{HV}</math>, i.e. usable for the DCO part of the PLL. This is checked by the Oscillator Watchdog of the PLL using the backup clock <math>f_{BACK}</math>. By using the crystal's nominal frequency (<math>f_{oscnom}</math>), the upper threshold calculates as follows:</p> <p><b>Note:</b> <math>f_{HV} = f_{oscnom} * 1,04 + 0,29</math></p> <ul style="list-style-type: none"> <li><math>0_B</math> The OSC frequency is not usable. Frequency <math>f_{osc}</math> is too high.</li> <li><math>1_B</math> The OSC frequency is usable</li> </ul>
<b>HYSEN</b>	9	rw	<p><b>Hysteresis Enable</b>  <math>0_B</math> Hysteresis is disabled  <math>1_B</math> Hysteresis is enabled (recommended)</p>
<b>HYSCTL</b>	11:10	rw	<p><b>Hysteresis Control</b>  <math>00_B</math> Hysteresis setting 1 (highest hysteresis, default)  <math>01_B</math> Hysteresis setting 2; Reserved for future use  <math>10_B</math> Hysteresis setting 3; Reserved for future use  <math>11_B</math> Hysteresis setting 4; Reserved for future use</p>

**Clocking System**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>AMPCTL</b>	13:12	rw	<p><b>Amplitude Control</b></p> <p>00<sub>B</sub> Amplitude control setting 1 (default value)      01<sub>B</sub> Amplitude control setting 2; Reserved for future use      10<sub>B</sub> Amplitude control setting 3; Reserved for future use      11<sub>B</sub> Amplitude control setting 4; Reserved for future use</p>
<b>OSCVAL</b>	20:16	rw	<p><b>OSC Frequency Value</b></p> <p>This bit field defines the divider value that generates the reference clock that is supervised by the oscillator watchdog. <math>f_{osc} = OSCCON.OSCVAL - 1 + 16</math> MHz.</p>
<b>APREN</b>	23	rw	<p><b>Amplitude Regulation Enable</b></p> <p>This bit field enables and disables Amplitude Regulation mode. When enabled, the bit field GAINSEL limits the maximum gain.</p> <p>0<sub>B</sub> Amplitude Regulation is disabled      1<sub>B</sub> Amplitude Regulation is enabled</p>
<b>CAP0EN</b>	24	rw	<p><b>Capacitance 0 Enable</b></p> <p><i>Note:</i> Total capacitance for each XTAL1 and XTAL2 is the sum of the enabled capacitance 0 to 3.</p> <p>0<sub>B</sub> Capacitance <math>C_{L0}</math> is disabled      1<sub>B</sub> Capacitance <math>C_{L0}</math> is enabled</p>
<b>CAP1EN</b>	25	rw	<p><b>Capacitance 1 Enable</b></p> <p><i>Note:</i> Total capacitance for each XTAL1 and XTAL2 is the sum of the enabled capacitance 0 to 3.</p> <p>0<sub>B</sub> Capacitance <math>C_{L1}</math> is disabled      1<sub>B</sub> Capacitance <math>C_{L1}</math> is enabled</p>
<b>CAP2EN</b>	26	rw	<p><b>Capacitance 2 Enable</b></p> <p><i>Note:</i> Total capacitance for each XTAL1 and XTAL2 is the sum of the enabled capacitance 0 to 3.</p> <p>0<sub>B</sub> Capacitance <math>C_{L2}</math> is disabled      1<sub>B</sub> Capacitance <math>C_{L2}</math> is enabled</p>
<b>CAP3EN</b>	27	rw	<p><b>Capacitance 3 Enable</b></p> <p><i>Note:</i> Total capacitance for each XTAL1 and XTAL2 is the sum of the enabled capacitance 0 to 3.</p> <p>0<sub>B</sub> Capacitance <math>C_{L3}</math> is disabled      1<sub>B</sub> Capacitance <math>C_{L3}</math> is enabled</p>
<b>0</b>	0, 15:14, 22:21, 31:28	r	<p><b>Reserved</b></p> <p>Read as 0; Should be written with 0.</p>

## Clocking System

**Table 279 Reset Values of OSCCON**

Reset Type	Reset Value	Note
System Reset	0000 0X1X <sub>H</sub>	

### 10.3.1.4 Configuration of the Oscillator

A configuration of the oscillator is always required before an external crystal / ceramic resonator can be used as clock source.

After any power-on reset the oscillator is disabled and needs to be configured as described in this section. During and after any other reset the oscillator is not affected and operates as previously configured, no re-configuration is required for this case.

For this start-up configuration, two options are supported:

- Configuration via the SSW
- Configuration after the execution of the SSW

#### Configuration via SSW

This option is enabled when bit FLASH0\_PROCOND.OSCCFG is set. In this mode the control information for the register OSCCON is loaded from FLASH0\_PROCOND by the SSW. Therefore the required information needs to be stored in UCB\_DFlash at offset 00<sub>H</sub>.

In this mode OSCCON.MODE, OSCCON.GAIN, OSCCON.HYSEN, OSCCON.HYSCTL, OSCCON.AMPCTL, and OSCCON.CAPxEN together with bit OSCCON.APREN are controllable.

If the oscillator was disabled it must be enabled by setting bit field OSCCON.MODE = 00<sub>B</sub>.

The gain setting should be adjusted to the needs of the connected external crystal / resonator in conjunction with the used capacitor configuration.

If the integrated capacitors should be used this can be enabled via the bits OSCCON.CAPxEN. If OSCCON[27:24] ≠ 0000<sub>B</sub> then bit OSCCON.APREN need to be set too. Please note that Amplitude Regulation must always be enabled when integrated caps are used. Enabling Amplitude Regulation does not require a change for OSCCON.GAINSEL, which should be left at 11<sub>B</sub>.

#### Configuration after SSW

After the optional configuration in the SSW there is always an option to configure the oscillator in the application software.

This is done via register OSCCON. The register itself is Safety ENDINIT protected.

In general the same rules apply for configuration as described in the previous section.

### 10.3.1.5 Oscillator Watchdog

The oscillator clock is selected as the source for the watchdog by configuring SYSPLLCON0.INSEL = 01<sub>B</sub>.

In combination with the System PLL, a monitoring function is implemented. This feature is defined to detect severe malfunctions of an external crystal / ceramic resonator. The system can detect a loss of clock input or a much too high input frequency (operation on a higher harmonic).

The oscillator watchdog monitors the incoming clock frequency  $f_{osc}$  from OSC. A stable and defined input frequency is a mandatory requirement for operation. Therefore this mode is selected automatically after each System Reset.

## Clocking System

The expected input frequency  $f_{\text{osc}}$  is selected via the bit field OSCCON.OSCVAL. The OSC\_WDT checks for frequencies which are too low or too high.

$$f_{\text{osc}} = \text{OSCCON.OSCVAL} - 1 + 16 \text{ MHz} \quad (10.1)$$

Before configuring the OSC\_WDT function, all the SMU Oscillator Watchdog alarm response options should be disabled, to avoid unintended SMU alarms. Thereafter the value of OSCCON.OSCVAL can be changed. Then the OSC\_WDT should be reset by setting OSCCON.OSCRES. This requests the start of OSC\_WDT monitoring with the new configuration. When the expected positive monitoring results of OSCCON.PLLLV and / or OSCCON.PLLHV are set, the input frequency is within the expected range. As setting OSCCON.OSCRES clears both bits OSCCON.PLLLV and OSCCON.PLLHV both status flags shall be set. Therefore both flags should be cleared before the SMU alarm responses are enabled again. The SMU alarm disabling-clearing-enabling sequence should also be used if only bit OSCCON.OSCRES is set without any modification of OSCCON.OSCVAL.

If the SMU detects an oscillator watchdog alarm, the same recovery procedure as from a PLL loss of lock event has to be executed.

**Note:** *The oscillator watchdog is intended to be used primarily when the PLL input clock  $f_{\text{osc}_i}$  is set to  $f_{\text{osc}_0}$  via register SYSPLLCON0.INSEL = 01<sub>B</sub>. If the SYCLK is used as source for  $f_{\text{osc}_i}$  via SYSPLLCON0.INSEL = 10<sub>B</sub>, the user shall either restrict the SYCLK frequency to be in the same range as when using a crystal or disable the watchdog alarms in the SMU. When using the backup clock as input for  $f_{\text{osc}_i}$  via SYSPLLCON0.INSEL = 00<sub>B</sub>, the watchdog alarms need to be disabled as well. The usable crystal frequency range and the allowed frequency range when driving the SYCLK via its assigned GPIO input pin is listed in the datasheet.*

### 10.3.2 Back-up Clock

A back-up clock source is available as an alternate clock source. This clock source provides a stable but reliable clock source that can be used as clock for the system. It provides less accuracy than an external crystal or ceramic resonator. The back-up clock can not be enabled or disabled, or otherwise be controlled that could prevent its general operation. Therefore, no control bits are available beside selecting the backup-clock as source (CCUCON0.CLKSEL = 00<sub>B</sub> as clock source for the clock distribution and SYSPLLCON0.INSEL = 00<sub>B</sub> as clock source for the two PLLs).

## 10.4 Clock Speed Up-Scaling (PLLs)

Typical CPU operating speeds are about 10 times faster (or even faster) than the speed of the used crystal as clock source. Therefore an up-scaling of the clock frequency is required.

For the up-scaling two Phase Lock Loop (PLLs) are provided.

### 10.4.1 System Phase-Locked Loop (System PLL) Module

The System PLL can convert a low-frequency external clock signal to a high-speed internal clock for maximum performance. It allows the use of a wide range of input and output frequencies by varying the different divider factors.

The System PLL also has fail-safe logic that detects degenerate external clock behavior, such as abnormal frequency deviations or a total loss of the external clock. It can execute emergency actions if it loses its lock on the external clock.

#### 10.4.1.1 Features

- DCO lock detection

---

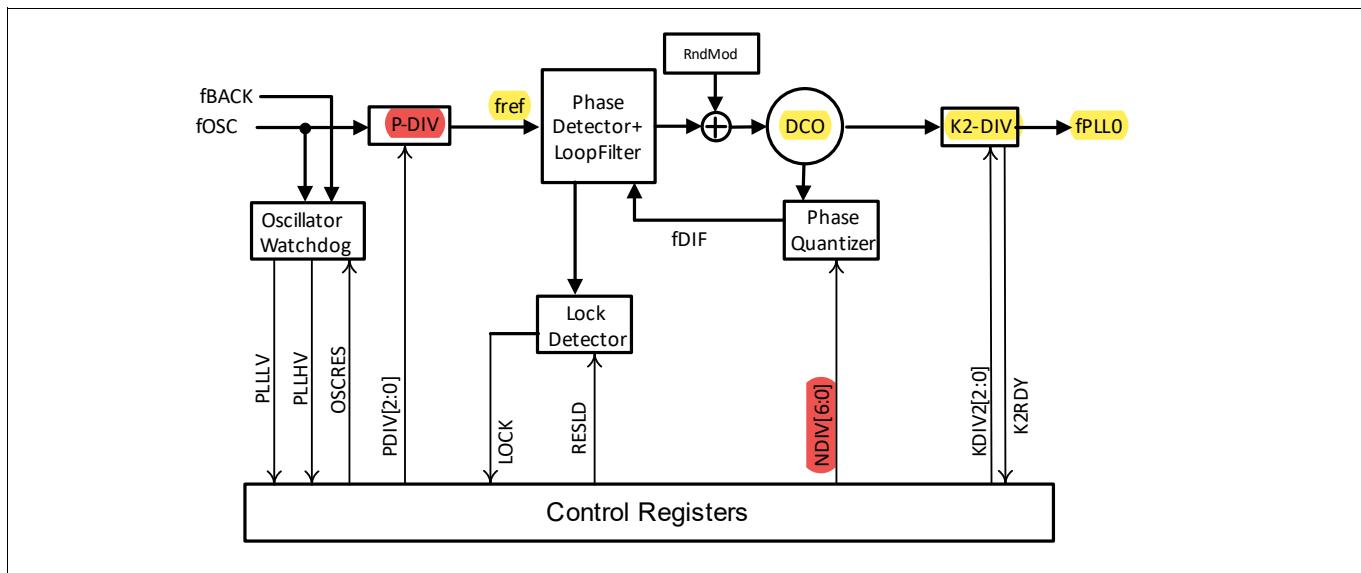
## Clocking System

- 3-bit input divider **P** (divide by PDIV+1)
- 7-bit feedback divider **N** (multiply by NDIV+1)
- 3-bit output divider **K2** (divide by K2DIV+1)
- Oscillator Watchdog
  - Detection of input frequencies that are too low
  - Detection of input frequencies that are too high
- Frequency Modulation with low jitter

## Clocking System

### 10.4.1.2 System PLL Functional Description

The following figure shows the System PLL block structure.



**Figure 85 System PLL Block Diagram**

#### Basic System PLL Operation

The input frequency  $f_{OSC}$  is divided down by a factor P, multiplied by a factor N, and then divided down by a factor K2.

The output frequency is given by

$$f_{PLL0} = (N * f_{OSC}) / (P * K2) \quad (10.2)$$

1)

Operation requires an input clock frequency of  $f_{OSC}$ . Therefore it is recommended to check and monitor if an input frequency  $f_{OSC}$  is available at all by checking OSCCON.PLLV. For better monitoring, the upper frequency can also be monitored via OSCCON.PLLHV.

The system operation frequency is controlled by the values of the three dividers: P, N, and K2. **A modification of the two dividers P and N has a direct influence to the DCO frequency and could lead to a loss of the Lock status.** A modification of the K2-divider has no impact on the lock status, but still changes the System PLL output frequency  $f_{PLL0}$ .

**Note:** *Changing the system operation frequency by changing the value of the K2-Divider has a direct coupling to the power consumption of the device. Therefore this has to be done carefully.*

When the frequency of the System PLL output frequency must be modified, the following sequence should be followed:

The SMU alarm generation for the Loss of Lock should be disabled.

While a different clock source is used for the CCU, the System PLL can be configured and checked for a positive DCO Lock status. The first target frequency should be selected in a way that it matches or is only slightly higher as the one used currently by the CCU. This avoids big changes in the system operation frequency (and therefore power consumption) when later switching to the System PLL. The P and N divider should be selected in the following way:

1) Please refer to the Data Sheet for valid PLL frequencies to determine the N,P and K2 settings

## Clocking System

- Select P and N in a way that  $f_{DCO}$  is in the lower area of its allowed values. This leads to slightly reduced power consumption but to slightly increased jitter.
- Select P and N in a way that  $f_{DCO}$  is in the upper area of its allowed values. This leads to slightly increased power consumption but to slightly reduced jitter.

After the P, N, and K2 dividers have been updated on the first configuration, the indication of the DCO Lock status should be checked (SYSPLLSTAT.LOCK = 1).

### Notes

1. It is recommended to reset the DCO Lock detection (SYSPLLCON0.RESLD = 1) after the new values of the dividers are configured to get a defined DCO lock check time.

After System PLL lock, the switch to the System PLL can be done.

The SMU status flag for the System PLL Loss-of-Lock event should be cleared and then enabled again. The intended System PLL output target frequency can now be configured by changing only the K2-Divider.

Depending on the selected value of the K2-Divider, the cycle time of the output clock is selected. This can have an impact for the operation with an external communication interface.

Change the K2-Divider in multiple steps, to avoid large changes in output frequency, and thus large changes in power consumption. Wait 6 cycles of  $f_{PLL0}$  between updates of the K2-Divider value.

### System PLL Lock Detection

The System PLL has a lock detection feature, that supervises the DCO part of the System PLL to differentiate between stable and unstable DCO circuit behavior. The lock detector marks the DCO circuit and therefore the output  $f_{DCO}$  of the DCO as unstable if the two inputs  $f_{REF}$  and  $f_{DIV}$  differ too much. Changes in one or both input frequencies below a certain level are not marked by a loss of lock, because the DCO can handle small changes without any problem for the system.

### System PLL Loss-of-Lock Event

The System PLL may become unlocked, due to a break of the crystal / ceramic resonator or the external clock line. In such a case, an SMU alarm event is generated.

### System PLL Power Down Mode

The System PLL offers a Power Down Mode. This mode can be entered to save power if the System PLL is not needed at all. The Power Down Mode is entered by setting bit SYSPLLCON0.PLLPWD. While the System PLL is in Power Down Mode, no System PLL output frequency is generated.

### Frequency Modulation

The System PLL output frequency  $f_{PLL0}$  can additionally be modified by a low-frequency modulation to reduce EMI. A random sequence is added to the DCO resulting in a randomly modulated  $f_{DCO}$ . The modulation frequency is defined by  $f_{REF}$ .

The modulation is enabled via bit SYSPLLCON0.MODEN. The modulation itself alters the DCO frequency randomly within the range of the configured modulation amplitude (MA). The modulation amplitude is selected via SYSPLLCON2.MODCFG[9:0].

$$\text{SYSPLLCON2.MODCFG[9:0]} = \text{HEX}[(64 * (\text{MA} / 100)) * (f_{osc} / P) * (N / f_{MV})] \quad (10.3)$$

Example: for MA = 1.25%;  $f_{osc}$  = 20 MHz; P = 2; N = 60;  $f_{MV}$  = 3.6 MHz the resulting bit field setting is 0x85.

The modulation is performed in a way that the resulting accumulated jitter added by the modulation stays below  $J_{MOD}$  (see the Data Sheet for the defined value). The modulation itself is monitored with  $f_{REF}$  and therefore the P-Divider should be configured with the smallest possible value.

**Clocking System****10.4.1.3 System PLL Registers**

The System PLL registers can be accessed by all CPUs in the system. However, it is suggested that only one CPU is used to control the clocks. As CPU0 is the active and available CPU immediately after a reset, this is the logical choice.

**System PLL Status Register**

These registers reflects the settings of the System PLL.

**SYSPLLSTAT****System PLL Status Register**(0014<sub>H</sub>)Reset Value: [Table 280](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								MODR UN	0	K2RDY	0	LOCK	PWDS TAT	0	r
r								rh	r	rh	r	rh	rh	r	

Field	Bits	Type	Description
PWDSTAT	1	rh	<b>System PLL Power-saving Mode Status</b> 0 <sub>B</sub> System PLL Power-saving Mode was not entered 1 <sub>B</sub> System PLL Power-saving Mode was entered
LOCK	2	rh	<b>System PLL Lock Status</b> <i>Note:</i> In case of a loss of lock, the $f_{DCO}$ is kept on the previous constant frequency. 0 <sub>B</sub> The frequency of the System PLL is not stable and doesn't enable system operation 1 <sub>B</sub> The frequency of the System PLL is stable and enables system operation
K2RDY	5	rh	<b>K2 Divider Ready Status</b> This bit indicates whether the K2-divider operates on the configured value. This is of interest when the SYSPLLCON1.K2DIV value is changed. <i>Note:</i> The PLL must be enabled and clocked to set the K2RDY field. 0 <sub>B</sub> K2-Divider does not yet operate with the new value 1 <sub>B</sub> K2-Divider operating with the new value
MODRUN	7	rh	<b>Modulation Run</b> This bit indicates if the frequency modulation of the System PLL is activated or not. 0 <sub>B</sub> Frequency modulation is not active 1 <sub>B</sub> Frequency modulation is active

## Clocking System

Field	Bits	Type	Description
0	0, 4:3, 6, 31:8	r	<b>Reserved</b> Read as 0.

**Table 280 Reset Values of SYSPLLSTAT**

Reset Type	Reset Value	Note
System Reset	0000 0002 <sub>H</sub>	

### System PLL Configuration 0 Register

#### SYSPLLCONO

**System PLL Configuration 0 Register (0018<sub>H</sub>) System Reset Value: 4000 3A00<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>INSEL</b>			<b>0</b>			<b>PDIV</b>				<b>0</b>			<b>RESLD</b>	<b>0</b>	<b>PLLPWD</b>
rw			r			rw				r			w	r	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													<b>MODE</b>	<b>0</b>	
													<b>N</b>		
													rw		rw

Field	Bits	Type	Description
<b>MODEN</b>	2	rw	<b>Modulation Enable</b> This bit controls the activation of the frequency modulation of the System PLL. 0 <sub>B</sub> Frequency modulation is not activated 1 <sub>B</sub> Frequency modulation is activated
<b>NDIV</b>	15:9	rw	<b>N-Divider Value</b> The value the N-Divider operates with is NDIV+1.
<b>PLLPWD</b>	16	rw	<b>System PLL Power Saving Mode</b>  <i>Note:</i> If the PLL has been powered down and is getting re-enabled via PLLPWD = 1, a wait period of 1 ms has to be applied until it is stable without jitter.  0 <sub>B</sub> The complete System PLL block is put into a Power Saving Mode and can no longer be used. 1 <sub>B</sub> Normal behavior
<b>RESLD</b>	18	w	<b>Restart DCO Lock Detection</b> Setting this bit will clear bit SYSPLLSTAT.LOCK and restart the DCO lock detection. Reading this bit returns always a zero.

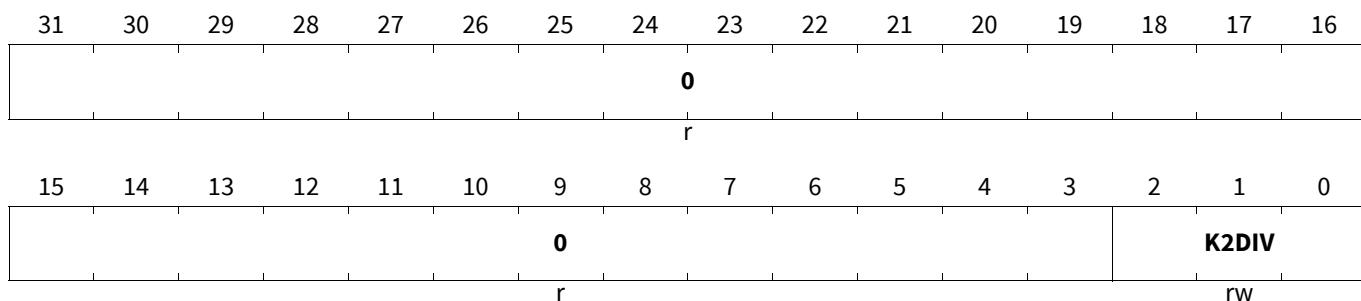
## Clocking System

Field	Bits	Type	Description
<b>PDIV</b>	26:24	rw	<b>P-Divider Value</b> The value the P-Divider operates with is PDIV+1.
<b>INSEL</b>	31:30	rw	<b>Input Selection</b> This bit field defines as clock source for the two PLLs (SystemPLL and Peripheral PLL). 00 <sub>B</sub> back-up clock is used as clock source 01 <sub>B</sub> $f_{OSC0}$ is used as clock source 10 <sub>B</sub> SYSCLK pin is used as clock source 11 <sub>B</sub> Reserved, do not use this combination
<b>0</b>	1:0, 8:3	rw	<b>Reserved</b> Read as 0; Should be written with 0.
<b>0</b>	17, 23:19, 29:27	r	<b>Reserved</b> Read as 0; Should be written with 0.

## System PLL Configuration 1 Register

### SYSPLLCON1

**System PLL Configuration 1 Register** **(001C<sub>H</sub>)** **System Reset Value: 0000 0005<sub>H</sub>**



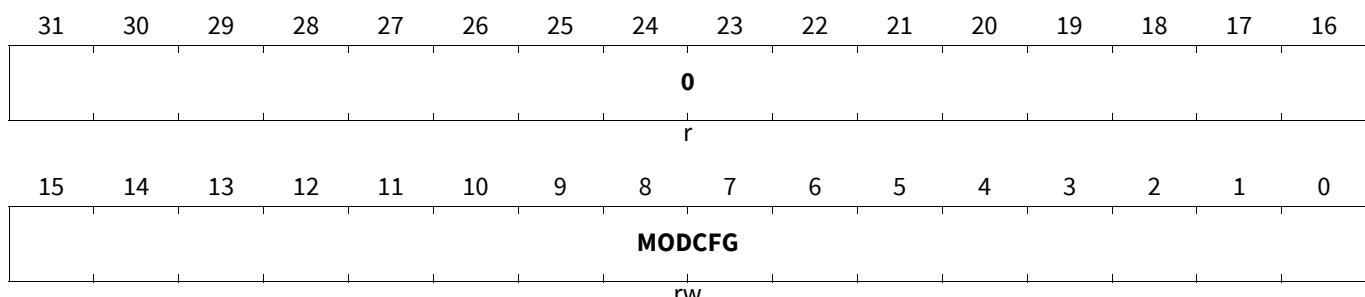
Field	Bits	Type	Description
<b>K2DIV</b>	2:0	rw	<b>K2-Divider Value</b> The value the K2-Divider operates with is K2DIV+1. While SYSPLLSTAT.K2RDY = 0, K2DIV is locked.
<b>0</b>	31:3	r	<b>Reserved</b> Read as 0; Should be written with 0.

## Clocking System

### System PLL Configuration 2 Register

#### SYSPLLCON2

**System PLL Configuration 2 Register (0020<sub>H</sub>) System Reset Value: 0000 6000<sub>H</sub>**



Field	Bits	Type	Description
<b>MODCFG</b>	15:0	rw	<b>Modulation Configuration</b> This bit field defines the modulation. MODCFG[9:0] defines the modulation amplitude. Bits MODCFG[9:5] are treated as integer part and bits MODCFG[4:0] as fractional part. Bits MODCFG[15:10] have to be configured with the following setting: 0x111101 <sub>B</sub> .
<b>0</b>	31:16	r	<b>Reserved</b> Read as 0; Should be written with 0.

### 10.4.2 Peripheral Phase-Locked Loop (Peripheral PLL) Module

The Peripheral PLL can convert a low-frequency external clock signal to a high-speed internal clock for maximum performance. It allows the use of input and output frequencies over a wide range by varying the different divider settings.

The Peripheral PLL also has fail-safe logic that detects degenerate external clock behavior such as abnormal frequency deviations or a total loss of the external clock. It can execute emergency actions if it loses its lock on the external clock.

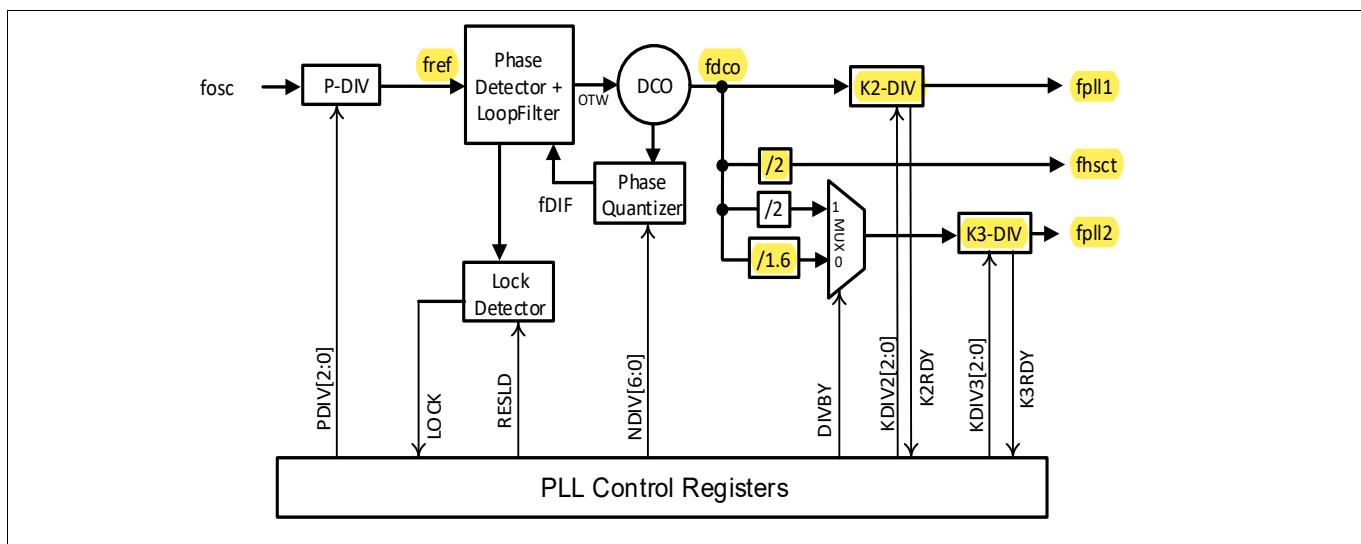
#### 10.4.2.1 Features

- DCO lock detection
- 3-bit input divider **P** (divide by PDIV+1)
- 7-bit feedback divider **N** (multiply by NDIV+1)
- 3-bit output divider **K2** (divide by K2DIV+1)
- 3-bit output divider **K3** (divide by K3DIV+1)

## Clocking System

### 10.4.2.2 Peripheral PLL Functional Description

The following figure shows the Peripheral PLL block structure.



**Figure 86 Peripheral PLL Block Diagram**

#### Basic Peripheral PLL Operation

The input frequency  $f_{osc}$  is divided down by a factor P, multiplied by a factor N, and then divided down by a factor K2 / K3.

The output frequency is given by:

$$f_{PLL1} = (N * f_{osc}) / (P * K2) \quad (10.4)$$

$$f_{PLL2} = (N * f_{osc}) / (P * K3 * 1,6) \text{ if } DIVBY = 0 \text{ or } f_{PLL2} = (N * f_{osc}) / (P * K3 * 2) \text{ if } DIVBY = 1 \quad (10.5)$$

1)

In addition the Peripheral PLL provides a special clock output  $f_{HSCT}$

$$f_{HSCT} = f_{Dco} / 2 \quad (10.6)$$

Operation does require an input clock frequency of  $f_{osc}$ . Therefore, it is recommended to check and monitor if an input frequency  $f_{osc}$  is available at all by checking OSCCON.PLLLV. For better monitoring, the upper frequency can also be monitored via OSCCON.PLLHV.

The system operation frequency is controlled by the values of the four dividers: P, N, and K2 / K3. A modification of the two dividers P and N has a direct influence to the DCO frequency and could lead to a loss of the Lock status. A modification of the K2 / K3-divider has no impact on the lock status, but still changes the Peripheral PLL output frequency  $f_{PLL1/2}$ .

**Note:** Changing the system operation frequency by changing the value of the K2 / K3-Divider has a direct coupling to the power consumption of the device. Therefore this has to be done carefully.

When the frequency of the Peripheral PLL output frequency must be modified, the following sequence should be followed:

The SMU alarm generation for Loss of Lock should be disabled.

1) Pls. refer to the Data Sheet for valid PLL frequencies to determine the N,P and K2/3 settings

## Clocking System

Configuration of the Peripheral PLL can be done independent of the System PLL. While configuring the Peripheral PLL the connected modules should either operate on the different clock source, or be configured for no operation. The P and N divider should be selected in the following way:

- Select P and N in a way that  $f_{DCO}$  is in the lower area of its allowed values. This leads to slightly reduced power consumption but to slightly increased jitter
- Select P and N in a way that  $f_{DCO}$  is in the upper area of its allowed values. This leads to slightly increased power consumption but to slightly reduced jitter

After the P, N, and K2 / K3 dividers have been updated on the first configuration, the indication of the Lock status should be checked (PERPLLSTAT.LOCK = 1).

### Notes

1. It is recommended to reset the Lock detection (PERPLLCON0.RESLD = 1) after the new values of the dividers are configured to get a defined lock check time.

After the Peripheral PLL is locked, the switch to the Peripheral PLL can be performed.

The SMU status flag for the Peripheral PLL Loss-of-Lock event should be cleared and then enabled again. The intended Peripheral PLL output target frequency can now be configured by changing only the K2 / K3-Divider.

Depending on the divider value of the K2 / K3-Divider the cycle time of the clock is selected. This can have an impact for the operation with an external communication interface.

Change the K2 / K3-Divider in multiple steps, to avoid large frequency changes, and thus large changes in power consumption. Wait 6 cycles of  $f_{PLL1}$  between updates of two K2 / K3-Divider.

### Peripheral PLL Lock Detection

The Peripheral PLL has a lock detection feature that supervises the DCO part of the Peripheral PLL in order to differentiate between stable and unstable DCO circuit behavior. The lock detector marks the DCO circuit and therefore the output  $f_{DCO}$  of the DCO as unstable if the two inputs  $f_{REF}$  and  $f_{DIV}$  differ too much. Changes in one or both input frequencies below a certain level are not marked by a loss of lock, because the DCO can handle small changes without any problem for the system.

### Peripheral PLL Loss-of-Lock Event

The Peripheral PLL may become unlocked, due to a break of the crystal / ceramic resonator or the external clock line. In such a case, an SMU alarm event is generated.

### Peripheral PLL Power Down Mode

The Peripheral PLL offers a Power Down Mode. This mode can be entered to save power if the Peripheral PLL is not needed at all. The Power Down Mode is entered by setting bit PERPLLCON0.PLLPWD. While the Peripheral PLL is in Power Down Mode, no Peripheral PLL output frequency is generated.

**Clocking System****10.4.2.3 Peripheral PLL Registers**

The Peripheral PLL registers can be accessed by all CPUs in the system. However, it is suggested that only one CPU is used to control the clocks. As CPU0 is the active and available CPU immediately after a reset, this is the logical choice.

**Peripheral PLL Status Register**

These registers displays the setting of the Peripheral PLL.

**PERPLLSTAT**

**Peripheral PLL Status Register** **(0024<sub>H</sub>)** **System Reset Value: 0000 0002<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PWDSTAT	1	rh	<b>Peripheral PLL Power-saving Mode Status</b> 0 <sub>B</sub> Peripheral PLL Power-saving Mode was not entered 1 <sub>B</sub> Peripheral PLL Power-saving Mode was entered
LOCK	2	rh	<b>Peripheral PLL Lock Status</b> <i>Note:</i> In case of a loss of lock, the $f_{DCO}$ is kept on the previous constant frequency. 0 <sub>B</sub> The frequency of the Peripheral PLL is not stable and doesn't enable system operation 1 <sub>B</sub> The frequency of the Peripheral PLL is stable and enables system operation
K3RDY	4	rh	<b>K3 Divider Ready Status</b> This bit indicates whether the K3-divider operates on the configured value. This is of interest when the PERPLLCON1.K3DIV value is changed. <i>Note:</i> The PLL must be enabled and clocked to set the K3RDY field. 0 <sub>B</sub> K3-Divider does not yet operate with the new value 1 <sub>B</sub> K3-Divider operating with the new value

## Clocking System

Field	Bits	Type	Description
<b>K2RDY</b>	5	rh	<b>K2 Divider Ready Status</b> This bit indicates whether the K2-divider operates on the configured value. This is of interest when the PERPLLCON1.K2DIV value is changed.  <i>Note:</i> The PLL must be enabled and clocked to set the K2RDY field. 0 <sub>B</sub> K2-Divider does not yet operate with the new value 1 <sub>B</sub> K2-Divider operating with the new value
<b>0</b>	0, 3, 6, 31:7	r	<b>Reserved</b> Read as 0.

### Peripheral PLL Configuration 0 Register

#### PERPLLCON0

Peripheral PLL Configuration 0 Register (0028 <sub>H</sub> )																System Reset Value: 0000 3E00 <sub>H</sub>			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16				
																0	RESLD	0	PLLP WD

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	0	RESLD	0	PLLP WD
																r	rw	r	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
																NDIV	0	DIVBY	rw
																rw			

Field	Bits	Type	Description
<b>DIVBY</b>	0	rw	<b>Divider Bypass</b> 0 <sub>B</sub> The divide-by-1.6 block in front of the K3-Divider is not bypassed. The resulting divider factor in front of the K3-Divider is $f_{DCO} / 1.6$ . 1 <sub>B</sub> The divide-by-1.6 block in front of the K3-Divider is bypassed. Taking into account the fix by two divider in front the resulting divider factor in front of the K3-Divider is $f_{DCO} / 2$ .
<b>NDIV</b>	15:9	rw	<b>N-Divider Value</b> The value the N-Divider operates with is NDIV+1.
<b>PLLPWD</b>	16	rw	<b>Peripheral PLL Power Saving Mode</b>  <i>Note:</i> If the PLL has been powered down and is getting re-enabled via PLLPWD = 1, a wait period of 1 ms has to be applied until it is stable without jitter.  0 <sub>B</sub> The complete Peripheral PLL block is put into a Power Saving Mode and can no longer be used. 1 <sub>B</sub> Normal behavior

## Clocking System

Field	Bits	Type	Description
<b>RESLD</b>	18	w	<b>Restart DCO Lock Detection</b> Setting this bit will clear bit SYSPLLSTAT.LOCK and restart the DCO lock detection. Reading this bit returns always a zero.
<b>PDIV</b>	26:24	rw	<b>P-Divider Value</b> The value the P-Divider operates with is PDIV+1.
<b>0</b>	8:1	rw	<b>Reserved</b> Read as 0; Should be written with 0.
<b>0</b>	17, 23:19, 31:27	r	<b>Reserved</b> Read as 0; Should be written with 0.

### Peripheral PLL Configuration 1 Register

#### PERPLLCON1

Peripheral PLL Configuration 1 Register (002C <sub>H</sub> )																System Reset Value: 0000 0001 <sub>H</sub>			
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16																0			
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0																0	K2DIV		

Field	Bits	Type	Description
<b>K2DIV</b>	2:0	rw	<b>K2-Divider Value</b> The value the K2-Divider operates with is K2DIV+1. While PERPLLSTAT.K2RDY = 0, K2DIV is locked.
<b>K3DIV</b>	10:8	rw	<b>K3-Divider Value</b> The value the K3-Divider operates with is K3DIV+1. While PERPLLSTAT.K3RDY = 0, K3DIV is locked.
<b>0</b>	7:3, 31:11	r	<b>Reserved</b> Read as 0; Should be written with 0.

## Clocking System

### 10.5 Clock Distribution (CCU)

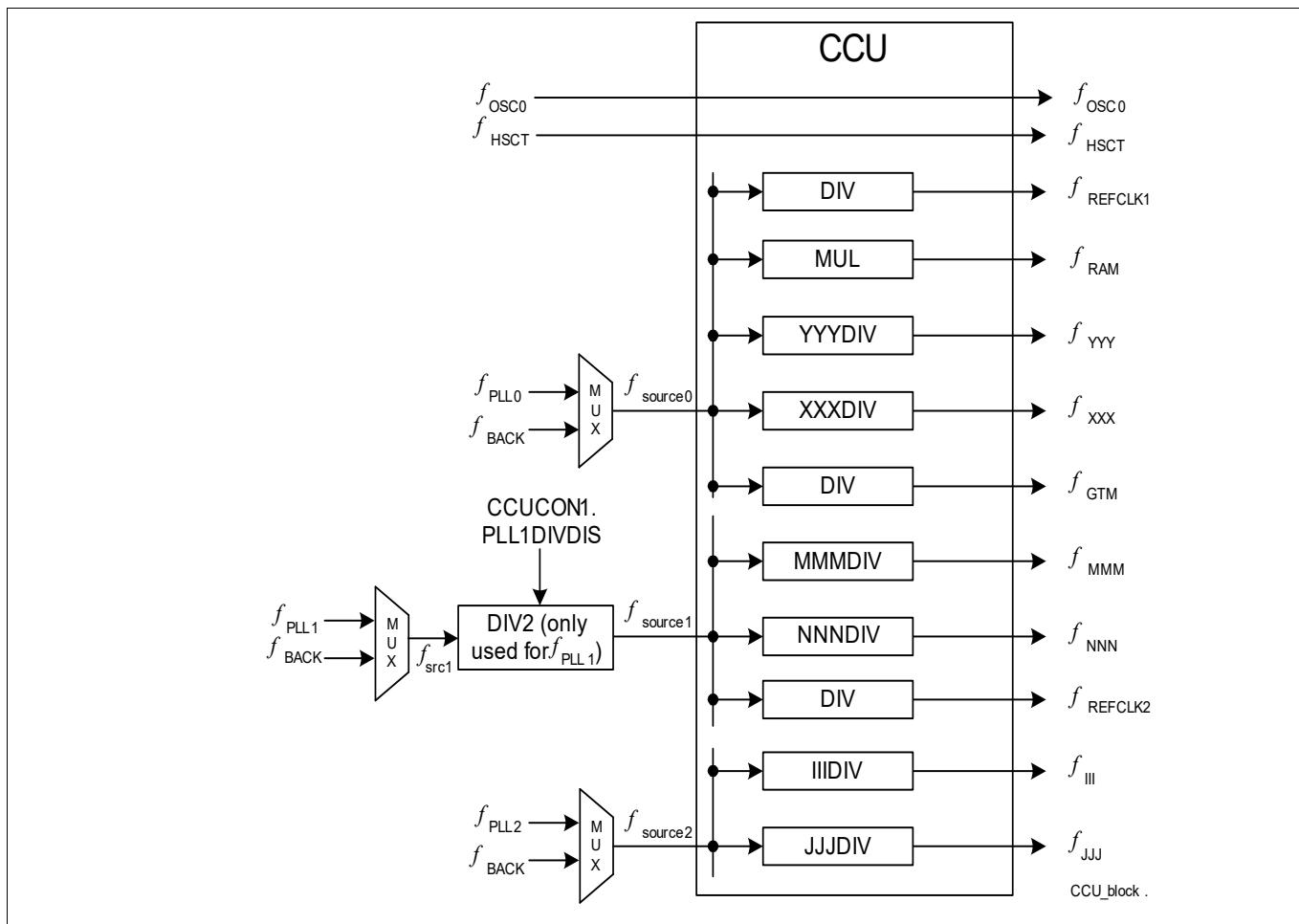
Using the first two parts of the Clock System, all root clocks the system relies on for operation are defined. In the following, these root clocks need to be individually adapted in frequency (divided) and distributed to all the MCU's modules, CPUs and blocks. This is done with focus on performance and power consumption optimization.

For clock distribution, the system is split into several sub-clock domains where the clock speed could be configured individually. There are also limitation for each sub-clock domain derived out of the internal interfaces. Each sub-clock domain defines a logical unit from a clocking perspective point of view.

The clock distribution is done via the Clock Control Unit (CCU). The CCU receives the clocks that are created by the two PLLs ( $f_{PLL0}$  and  $f_{PLL1/2}$ ), the back-up clock  $f_{Back}$ , and  $f_{Osc0}$ . These clocks are either forwarded directly or divided in order to supply the sub-clock domains.

#### 10.5.1 Clock Control Unit

The Clock Control Unit (CCU) receives the clocks that are created by the two PLLs ( $f_{PLL0}$  and  $f_{PLL1/2}$ ), the back-up clock  $f_{Back}$ , and  $f_{Osc0}$ .



**Figure 87 Clock Control Unit Overview**

For most clocks, a linear divider is provided to adapt the clock frequency to the application requirement. This divider is controlled by the bit field XXXDIV in the registers CCUCONy.

The clocking system is very flexible in its configuration. This allows the adaptation of the frequency change to a specific current limit defined by the application.

## Clocking System

For the GTM, there is a fixed divider and a programmable divider in the CCU, delivering a clock that is only dependent on the System PLL or the back-up clock.

$$f_{\text{GTM}} = f_{\text{SOURCEGTM}} / \text{GTMDIV} \quad (10.7)$$

$$f_{\text{SOURCEGTM}} = f_{\text{SPB}} * 2 \text{ if } \text{GTMDIV} = 0001_B \text{ else } f_{\text{SOURCEGTM}} = f_{\text{SOURCEO}} \quad (10.8)$$

For the Ethernet RAMs, there is a fixed divider in the CCU, delivering a constant clock from either the System PLL or the back-up clock.

$$f_{\text{RAM}} = f_{\text{SPB}} * 2 \quad (10.9)$$

For the ERAY, there is a fixed divider in the CCU, delivering a constant clock from either the Peripheral PLL  $f_{\text{PLL1}}$  or the back-up clock.

$$f_{\text{ERAY}} = f_{\text{source1}} / 2 \quad (10.10)$$

For the EBU, there is a fixed divider in the CCU, delivering a constant clock from either the Peripheral PLL  $f_{\text{PLL1}}$  or the back-up clock.

$$f_{\text{EBU}} = f_{\text{source1}} \quad (10.11)$$

For the ADCs, there is a fixed divider in the CCU, delivering a constant clock from either the Peripheral PLL  $f_{\text{PLL1}}$  or the back-up clock.

$$f_{\text{ADC}} = f_{\text{source1}} \quad (10.12)$$

For the HSPDM, there is a fixed divider in the CCU, delivering a constant clock from either the Peripheral PLL  $f_{\text{PLL1}}$  or the back-up clock.

$$f_{\text{HSPDM\_160}} = f_{\text{source1}} \quad (10.13)$$

$$f_{\text{HSPDM\_320}} = f_{\text{src1}} \quad (10.14)$$

For the HSCT there is a connection from Peripheral PLL  $f_{\text{PLL1}}$ .

$$f_{\text{HSCT}} = f_{\text{DCO}} / 2 \quad (10.15)$$

There is also a fixed reference clock REFCLK1/2 for the debug block, which divides the master clock  $f_{\text{SOURCEO/1}}$  by 24. This allows the OCDS to generate timestamps independent of the selected SRI and SPB clock speeds.

### 10.5.1.1 Basic Clock System Mechanisms

The clocking system offers many options and a high amount of flexibility. For different IPs (CPUs, modules or module clusters), options are included to influence the application execution / performance via the clock control.

The following is a brief overview of the different clocks:

- System buses
  - $f_{\text{SRI}}$  defines the operating performance of the SRI-Bus, and therefore the data exchange rate between all connected masters and slaves
  - $f_{\text{SPB}}$  defines the operating performance of the SPB-Bus, and therefore the data exchange rate between all connected masters and slaves and the interrupt system
- CPU controls
  - $f_{\text{CPUx}}$  defines the execution speed of CPUx
- PMU controls
  - $f_{\text{FSI2}}$  defines the execution speed of the PFlash for reading operations
  - $f_{\text{FSI}}$  defines the execution speed for all other flash operations

## Clocking System

- Peripheral options
  - $f_{STM}$  defines the basic frequency for the STMs independent of the rest of the system (beside the limitations listed in [Table 282](#)). This allows the STMs to operate on a constant frequency.
  - $f_{GTM}$  defines the basic frequency for the GTM independent of the rest of the system (beside the limitations listed in [Table 282](#)). This allows the GTM to operate on a constant frequency.
  - $f_{MCAN}$  defines the basic frequency for the MCAN independent of the rest of the system. This allows the MCAN to operate on a constant baud rate (frequency).
  - $f_{MCANH}$  defines the frequency for the internal clocking of the MCAN module. This frequency is independent to  $f_{SPB}$  and allows the MCAN to continue operation when  $f_{SPB}$  is reduced in frequency, thus enabling pretended networking.  $f_{MCANH}$  must always be greater than  $f_{MCAN}$ .
  - $f_{ADC}$  defines the basic frequency for the ADCs (valid for EVADC and EDSADC) independent of the rest of the system. This allows the ADCs to operate on a constant frequency.
  - $f_{MSC}$  defines the basic frequency for the MSCs independent of the rest of the system. This allows the MSCs to operate on a constant baud rate (frequency).
  - $f_{QSPI}$  defines the basic frequency for the QSPIs independent of the rest of the system. This allows the QSPIs to operate on a constant baud rate (frequency).
  - $f_{ASCLINF/S}$  defines the basic frequencies for the ASCLINs independent of the rest of the system. This allows the ASCLINs to operate on a constant baud rate (frequency).
  - $f_{EBU}$  defines the basic frequency for the EBU independent of the rest of the system. This allows the EBU to operate on a constant baud rate (frequency). Please note that this clock is not available in all products. Please refer to the Platform Feature Overview whether the module EBU is present at all for the variant you are using.
  - $f_{I2C}$  defines the basic frequency for the I2C independent of the rest of the system. This allows the I2C to operate on a constant baud rate (frequency).
  - $f_{ERAY}$  defines the basic frequency for the ERAY independent of the rest of the system. This allows the ERAY to operate on a constant baud rate (frequency).
  - $f_{HSCT}$  defines the basic frequency for the HSCT independent of the rest of the system. This allows the HSCT to operate on a constant baud rate (frequency).
  - $f_{GETH}$  defines the basic frequency for the Gigabit Ethernet Kernel. This frequency is independent of  $f_{SPB}$  and allows the Gigabit Ethernet to operate at a constant baud rate (frequency).
  - $f_{ADAS}$  defines the basic frequency for the SPU and RIF Kernels. This frequency is independent to  $f_{SRI}$  and allows the SPU and RIF to operate at a constant frequency when  $f_{SRI}$  is reduced in frequency. Please note that this clock is not available in all products. Please refer to the Platform Feature Overview whether the modules RIF and SPU are present at all for the variant you are using.
- Debug system
  - Because debugging should be non-intrusive to the application system, a separate clock  $f_{BBB}$  for dedicated debug resources is available. This allows debugging (trace generation) during changes of the other clock configurations. Please note that  $f_{BBB}$  needs to be faster than or equal to  $f_{SPB}$  for debug.

**Table 281 CCU Clock Options**

CCU Clock Output	Clock Source				
	System PLL ( $f_{PLL0}$ )	Peripheral PLL ( $f_{PLL1}$ )	Peripheral PLL ( $f_{PLL2}$ )	Back-up ( $f_{BACK}$ )	OSC_XTAL ( $f_{osco}$ )
$f_{SRI}$	✓	-	-	Default	-
$f_{CPUx}$	$f_{SRI}$	-	-	-	-

## Clocking System

**Table 281 CCU Clock Options (cont'd)**

CCU Clock Output	Clock Source				
	System PLL ( $f_{PLL0}$ )	Peripheral PLL ( $f_{PLL1}$ )	Peripheral PLL ( $f_{PLL2}$ )	Back-up ( $f_{BACK}$ )	OSC_XTAL ( $f_{osco}$ )
$f_{SPB}$	✓	-	-	Default	-
$f_{FSI}$	$f_{SRI}$	-	-	-	-
$f_{FSI2}$	$f_{SRI}$	-	-	-	-
$f_{REFCLK1}$	✓	-	-	Default	-
$f_{REFCLK2}$	-	✓	-	Default	-
$f_{BBB}$	✓	-	-	Default	-
$f_{ERAY}$	-	✓	-	-	-
$f_{GTM}$	✓	-	-	Default	-
$f_{STM}$	✓	-	-	Default	-
$f_{MSC}$	-	✓	✓	Default	-
$f_{GETH}$	✓	-	-	Default	✓
$f_{ADAS}$	✓	-	-	Default	✓
$f_{MCANH}$	✓	-	-	Default	✓
$f_{MCAN}$	-	✓	-	Default	✓
$f_{ASCLINF}$	-	-	✓	Default	-
$f_{ASCLINS}$	-	✓	-	Default	✓
$f_{QSPI}$	-	✓	✓	Default	-
$f_{ADC}$	-	✓	-	Default	-
$f_{I2C}$	-	-	✓	Default	-
$f_{EBU}$	-	✓	-	Default	-
$f_{HSPDM\_160}$	-	✓	-	Default	-
$f_{HSPDM\_320}$	-	✓	-	Default	-

The overall clock system is divided into two major parts, each operating on a clock derived from one of the two available PLLs. The two parts are:

- System and Compute block containing:
  - CPUs and DMAs with its associated memories
  - System infrastructure such as the busses, clocking, reset and power control, interrupt system
  - Timers
  - Dedicated communication interfaces (Ethernet, SENT, and PSI5(-S))
  - Debug system
- Application block containing
  - ADCs
  - Jitter-sensitive communication interfaces (ERAY, HSCT, QSPI, MSC, ASCLIN, MCAN, I2C, and EBU)

This separation allows an independent configuration of the desired baud rates and conversion rates, independent of the power-optimized configuration of the system.

## Clocking System

### 10.5.1.2 Clock Divider Limitations

For the clock dividers which control the different sub-clock domains / modules, the allowed values are limited and the following ratios have to be observed. The ratios are defined in the following way: clock A =  $f_{AAA}$ ; clock B =  $f_{BBB}$  the allowed ratio is 1 : n where n has a defined range of values. Clock A is always faster or equal to clock B in this definition with  $f_{AAA}$  [MHz] = n \*  $f_{BBB}$  [MHz]. For synchronous clocks, n must be an integer value.

In addition, the divider values are limited by the resulting minimum/maximum frequencies. These frequencies are defined in the Target Data Sheet / ACDC Target Specification.

**Table 282 CCU allowed Clock Ratios**

Clock A	Clock B	Allowed Ratios	Notes	Recommended Default
$f_{SRI}$	$f_{SPB}$	1 : n	n = 1, 2, 3, 4, 5, 6,...	n = 2 or 3
$f_{SRI}$	$f_{FSI}$	1 : n	n = 1, 2, 3	n = 2 or 3
$f_{FSI2}$	$f_{FSI}$	1 : n	n = 1, 2, 3	n = 2 or 3
$f_{FSI}$	$f_{SPB}/2^1)$	1 : n	n = 1, 2, 3, 4, 5, 6,...	n = 2
$f_{GTM}$	$f_{SPB}$	1 : n	n = 1, 2	n = 2
$f_{SPB}$	$f_{GTM}$	1 : n	n = 1, 2, 3, 4, 5, 6	n = 2
$f_{SPB}$	$f_{STM}$	1 : n <sup>2)</sup>	n = 1, 2, 3, 4, 5, 6	n = 1
$f_{STM}$	$f_{SPB}$	1 : n <sup>2)</sup>	n = 1, 2, 3, 4, 5, 6,...	n = 1
$f_{SRI}$	$f_{BBB}$ <sup>3)</sup>	1 : n	n = 1, 2	n = 2
$f_{SRI}$	$f_{GETH}$	1 : n	n = 1, 2, 3, 4, 5, 6,...	n = 2
$f_{SPB}$	$f_{MCANH}$	1 : n	n = 1 <sup>4)</sup>	n = 1
$f_{MCANH}$	$f_{MCAN}$	1 : n <sup>5)</sup>	n >= 1.0	n = 1.0
$f_{ADAS}$	$f_{SRI}$	1 : n	n = 1, 2	n = 1
$f_{ADAS}$	$f_{BBB}$	1 : n <sup>6)</sup>	n = 2	n = 2

1) for FSI module, SPB-half clock is internally used, hence relevant for clock ratios

2)  $f_{STM}$  can be faster, slower, or equal to  $f_{SPB}$

3)  $f_{BBB}$  has to be slower, or equal to  $f_{SRI}$

4) restriction only valid when not in pretended networking / LP mode

5) n is not an integer as the related clocks are asynchronous to each other

6)  $f_{BBB}$  has to be half the SPU frequency  $f_{ADAS}$

**Note:** The recommended values do not necessarily reflect the default configuration after a reset event. Instead they should give a hint about how to configure the system for the optimal performance. In addition, it may happen that applying an allowed ratio on two selected clocks violates the ratio of others. Due to the complexity, this cannot be shown here in all combinations. The user has to take care not to violate clock ratios depending on the use case, i.e. enabled clocks.

## Clocking System

### 10.5.1.3 CCU Registers

The CCU registers may be accessed by any CPU in the system. However, it is suggested that only one CPU is used to control the clocks. As CPU0 is the active and available CPU after a reset, this is the logical choice.

#### CCU Clock Control Register 0

**CCUCONO**

**CCU Clock Control Register 0**

(0030<sub>H</sub>)

**Reset Value: Table 283**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	<b>UP</b>	<b>CLKSEL</b>		<b>FSI2DIV</b>		<b>FSIDIV</b>			<b>BBBDIV</b>			<b>SPBDIV</b>			
rh	w	rwh		rw		rw			rw			rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>0</b>		<b>LPDIV</b>			<b>SRIDIV</b>			<b>GTMDIV</b>			<b>STMDIV</b>				
rw		rw			rw				rw			rw			

Field	Bits	Type	Description
<b>STMDIV</b>	3:0	rw	<p><b>STM Divider Reload Value</b></p> <p>The resulting STM frequency is configured to <math>f_{STM} = f_{source0} / STMDIV</math> for the allowed configurations. For STMDIV = 0000<sub>B</sub> the clock is shut off. <math>f_{source0}</math> can be configured either to <math>f_{PLL0}</math> (CLKSEL = 01<sub>B</sub>) or <math>f_{BACK}</math> (CLKSEL = 00<sub>B</sub>)</p> <p> <math>0_H</math> <math>f_{STM}</math> is stopped  <math>1_H</math> <math>f_{STM} = f_{source0}</math>  <math>2_H</math> <math>f_{STM} = f_{source0}/2</math>  <math>3_H</math> <math>f_{STM} = f_{source0}/3</math>  <math>4_H</math> <math>f_{STM} = f_{source0}/4</math>  <math>5_H</math> <math>f_{STM} = f_{source0}/5</math>  <math>6_H</math> <math>f_{STM} = f_{source0}/6</math>  <math>7_H</math> Reserved, do not use this combination  <math>8_H</math> <math>f_{STM} = f_{source0}/8</math>  <math>9_H</math> Reserved, do not use this combination  <math>A_H</math> <math>f_{STM} = f_{source0}/10</math>  <math>B_H</math> Reserved, do not use this combination  <math>C_H</math> <math>f_{STM} = f_{source0}/12</math>  <math>D_H</math> Reserved, do not use this combination  <math>E_H</math> Reserved, do not use this combination  <math>F_H</math> <math>f_{STM} = f_{source0}/15</math> </p>

## Clocking System

Field	Bits	Type	Description
<b>GTMDIV</b>	7:4	rw	<p><b>GTM Divider Reload Value</b></p> <p>The resulting GTM frequency is configured to <math>f_{GTM} = f_{SOURCEGTM} / GTMDIV</math> for the allowed configurations. For GTMDIV = 0000<sub>B</sub> the clock is shut off.</p> <ul style="list-style-type: none"> <li>0<sub>H</sub> <math>f_{GTM}</math> is stopped</li> <li>1<sub>H</sub> <math>f_{GTM} = f_{SOURCEGTM}</math></li> <li>2<sub>H</sub> <math>f_{GTM} = f_{SOURCEGTM}/2</math></li> <li>3<sub>H</sub> <math>f_{GTM} = f_{SOURCEGTM}/3</math></li> <li>4<sub>H</sub> <math>f_{GTM} = f_{SOURCEGTM}/4</math></li> <li>5<sub>H</sub> <math>f_{GTM} = f_{SOURCEGTM}/5</math></li> <li>6<sub>H</sub> <math>f_{GTM} = f_{SOURCEGTM}/6</math></li> <li>7<sub>H</sub> Reserved, do not use this combination</li> <li>8<sub>H</sub> <math>f_{GTM} = f_{SOURCEGTM}/8</math></li> <li>9<sub>H</sub> Reserved, do not use this combination</li> <li>A<sub>H</sub> <math>f_{GTM} = f_{SOURCEGTM}/10</math></li> <li>B<sub>H</sub> Reserved, do not use this combination</li> <li>C<sub>H</sub> <math>f_{GTM} = f_{SOURCEGTM}/12</math></li> <li>D<sub>H</sub> Reserved, do not use this combination</li> <li>E<sub>H</sub> Reserved, do not use this combination</li> <li>F<sub>H</sub> <math>f_{GTM} = f_{SOURCEGTM}/15</math></li> </ul>
<b>SRIDIV</b>	11:8	rw	<p><b>SRI Divider Reload Value</b></p> <p>The resulting SRI frequency is configured to <math>f_{SRI} = f_{source0} / SRIDIV</math> for the allowed configurations.</p> <p><math>f_{source0}</math> could be configured either to <math>f_{PLL0}</math> (CLKSEL = 01<sub>B</sub>) or <math>f_{BACK}</math> (CLKSEL = 00<sub>B</sub>)</p> <ul style="list-style-type: none"> <li>0<sub>H</sub> Reserved, do not use this combination</li> <li>1<sub>H</sub> <math>f_{SRI} = f_{source0}</math></li> <li>2<sub>H</sub> <math>f_{SRI} = f_{source0}/2</math></li> <li>3<sub>H</sub> <math>f_{SRI} = f_{source0}/3</math></li> <li>4<sub>H</sub> <math>f_{SRI} = f_{source0}/4</math></li> <li>5<sub>H</sub> <math>f_{SRI} = f_{source0}/5</math></li> <li>6<sub>H</sub> <math>f_{SRI} = f_{source0}/6</math></li> <li>7<sub>H</sub> Reserved, do not use this combination;</li> <li>8<sub>H</sub> <math>f_{SRI} = f_{source0}/8</math></li> <li>9<sub>H</sub> Reserved, do not use this combination</li> <li>A<sub>H</sub> <math>f_{SRI} = f_{source0}/10</math></li> <li>B<sub>H</sub> Reserved, do not use this combination</li> <li>C<sub>H</sub> <math>f_{SRI} = f_{source0}/12</math></li> <li>D<sub>H</sub> Reserved, do not use this combination</li> <li>E<sub>H</sub> Reserved, do not use this combination</li> <li>F<sub>H</sub> <math>f_{SRI} = f_{source0}/15</math></li> </ul>

## Clocking System

Field	Bits	Type	Description
<b>LPDIV</b>	14:12	rw	<p><b>Low Power Divider Reload Value</b></p> <p><i>Note:</i> The selected divider is valid for all clocks derived from <math>f_{XXX}</math> with <math>XXX = SPB, SRI, BBB, FSI, GETH, GTM, ADAS</math>.</p> <p> <math>000_B</math> <math>f_{XXX}</math> controlled by the related CCUCON0/5 bit fields  <math>001_B</math> <math>f_{XXX} = f_{source0} / 30</math>  <math>010_B</math> <math>f_{XXX} = f_{source0} / 60</math>  <math>011_B</math> <math>f_{XXX} = f_{source0} / 120</math>  <math>100_B</math> <math>f_{XXX} = f_{source0} / 240</math>  <math>101_B</math> Reserved, do not use this combination  ...  <math>111_B</math> Reserved, do not use this combination </p>
<b>SPBDIV</b>	19:16	rw	<p><b>SPB Divider Reload Value</b></p> <p><math>f_{source0}</math> could be configured either to <math>f_{PLL}</math> (<math>CLKSEL = 01_B</math>) or <math>f_{BACK}</math> (<math>CLKSEL = 00_B</math>)</p> <p> <math>0_H</math> Reserved, do not use this combination  <math>1_H</math> Reserved, do not use this combination  <math>2_H</math> <math>f_{SPB} = f_{source0}/2</math>  <math>3_H</math> <math>f_{SPB} = f_{source0}/3</math>  <math>4_H</math> <math>f_{SPB} = f_{source0}/4</math>  <math>5_H</math> <math>f_{SPB} = f_{source0}/5</math>  <math>6_H</math> <math>f_{SPB} = f_{source0}/6</math>  <math>7_H</math> Reserved, do not use this combination  <math>8_H</math> <math>f_{SPB} = f_{source0}/8</math>  <math>9_H</math> Reserved, do not use this combination  <math>A_H</math> <math>f_{SPB} = f_{source0}/10</math>  <math>B_H</math> Reserved, do not use this combination  <math>C_H</math> <math>f_{SPB} = f_{source0}/12</math>  <math>D_H</math> Reserved, do not use this combination  <math>E_H</math> Reserved, do not use this combination  <math>F_H</math> <math>f_{SPB} = f_{source0}/15</math> </p>

## Clocking System

Field	Bits	Type	Description
<b>BBBDIV</b>	23:20	rw	<p><b>BBB Divider Reload Value</b></p> <p>The resulting BBB frequency is configured to <math>f_{\text{BBB}} = f_{\text{source0}} / \text{BBBDIV}</math> for all allowed configurations. For <math>\text{BBBDIV} = 0000_B</math> the clock is shut off.</p> <p><math>f_{\text{source0}}</math> could be configured either to <math>f_{\text{PLL0}}</math> (<math>\text{CLKSEL} = 01_B</math>) or <math>f_{\text{BACK}}</math> (<math>\text{CLKSEL} = 00_B</math>)</p> <ul style="list-style-type: none"> <li><math>0_H</math> <math>f_{\text{BBB}}</math> is stopped</li> <li><math>1_H</math> <math>f_{\text{BBB}} = f_{\text{source0}}</math></li> <li><math>2_H</math> <math>f_{\text{BBB}} = f_{\text{source0}}/2</math></li> <li><math>3_H</math> <math>f_{\text{BBB}} = f_{\text{source0}}/3</math></li> <li><math>4_H</math> <math>f_{\text{BBB}} = f_{\text{source0}}/4</math></li> <li><math>5_H</math> <math>f_{\text{BBB}} = f_{\text{source0}}/5</math></li> <li><math>6_H</math> <math>f_{\text{BBB}} = f_{\text{source0}}/6</math></li> <li><math>7_H</math> Reserved, do not use this combination</li> <li><math>8_H</math> <math>f_{\text{BBB}} = f_{\text{source0}}/8</math></li> <li><math>9_H</math> Reserved, do not use this combination</li> <li><math>A_H</math> <math>f_{\text{BBB}} = f_{\text{source0}}/10</math></li> <li><math>B_H</math> Reserved, do not use this combination</li> <li><math>C_H</math> <math>f_{\text{BBB}} = f_{\text{source0}}/12</math></li> <li><math>D_H</math> Reserved, do not use this combination</li> <li><math>E_H</math> Reserved, do not use this combination</li> <li><math>F_H</math> <math>f_{\text{BBB}} = f_{\text{source0}}/15</math></li> </ul>
<b>FSIDIV</b>	25:24	rw	<p><b>FSI Divider Reload Value</b></p> <ul style="list-style-type: none"> <li><math>00_B</math> Reserved, do not use this combination</li> <li><math>01_B</math> <math>f_{\text{FSI}} = f_{\text{SRI}}</math></li> <li><math>10_B</math> <math>f_{\text{FSI}} = f_{\text{SRI}} / 2</math> for <math>\text{SRIDIV} = 0001_B</math> or <math>0010_B</math>, else <math>f_{\text{FSI}} = f_{\text{SRI}}</math></li> <li><math>11_B</math> <math>f_{\text{FSI}} = f_{\text{SRI}} / 3</math> for <math>\text{SRIDIV} = 0001_B</math> or <math>0010_B</math>, else <math>f_{\text{FSI}} = f_{\text{SRI}}</math></li> </ul>
<b>FSI2DIV</b>	27:26	rw	<p><b>FSI2 Divider Reload Value</b></p> <ul style="list-style-type: none"> <li><math>00_B</math> Reserved, do not use this combination</li> <li><math>01_B</math> <math>f_{\text{FSI2}} = f_{\text{SRI}}</math></li> <li><math>10_B</math> Reserved, do not use this combination</li> <li><math>11_B</math> Reserved, do not use this combination</li> </ul>
<b>CLKSEL</b>	29:28	rwh	<p><b>Clock Selection for Source</b></p> <p>This bit field defines the clock source that is used for the clock generation of <math>f_{\text{sourcex}}</math>.</p> <ul style="list-style-type: none"> <li><math>00_B</math> <math>f_{\text{BACK}}</math> is used as clock source <math>f_{\text{source0}}</math>, <math>f_{\text{src1}}</math>, and <math>f_{\text{source2}}</math></li> <li><math>01_B</math> <math>f_{\text{PLL0}}</math> is used as clock source <math>f_{\text{source0}}</math>; <math>f_{\text{PLL1}}</math> is used as clock source <math>f_{\text{src1}}</math>; <math>f_{\text{PLL2}}</math> is used as clock source <math>f_{\text{source2}}</math></li> <li><math>10_B</math> Reserved, do not use this combination</li> <li><math>11_B</math> Reserved, do not use this combination</li> </ul>
<b>UP</b>	30	w	<p><b>Update Request</b></p> <p>Setting this bit will request an update for CCUCON0 and CCUCON5. Only one UP bit must be set for either CCUCON0 or CCUCON5. This bit always reads as zero.</p> <ul style="list-style-type: none"> <li><math>0_B</math> No action</li> <li><math>1_B</math> A new complete parameter set is transferred to the CCU defined by registers CCUCON0 and CCUCON5.</li> </ul>

## Clocking System

Field	Bits	Type	Description
LCK	31	rh	<p><b>Lock Status</b>            This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <p><i>Note:</i> <i>The lock bit is set when an update of CCUCON0/5 has been requested, and released when the update is complete.</i></p> <p>0<sub>B</sub> The register is unlocked and can be updated            1<sub>B</sub> The register is locked and can not be updated</p>
0	15	rw	<p><b>Reserved</b>            Should be written with 0.</p>

**Table 283 Reset Values of CCUCONO**

Reset Type	Reset Value	Note
After SSW execution	0512 0112 <sub>H</sub>	
System Reset	0313 0113 <sub>H</sub>	

### CCU Clock Control Register 1

**CCUCON1**

**CCU Clock Control Register 1**

(0034<sub>H</sub>)

**System Reset Value: 1010 0302<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	<b>0</b>	<b>CLKSELQSPI</b>		<b>QSPIDIV</b>			<b>0</b>		<b>CLKSELMSC</b>		<b>MSCDIV</b>				
rh	rw	rw		rw			rw		rw	rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		<b>0</b>			<b>I2CDIV</b>		<b>PLL1D</b>	<b>IVDIS</b>	<b>0</b>	<b>CLKSELMCAN</b>		<b>MCANDIV</b>			
		rw			rw		rw	rw	rw	rw		rw		rw	

## Clocking System

Field	Bits	Type	Description
<b>MCANDIV</b>	3:0	rw	<p><b>MCAN Divider Reload Value</b></p> <p>The resulting MCAN frequency is configured to <math>f_{\text{MCANI}} = f_{\text{source1}} / \text{MCANDIV}</math> for the allowed configurations. For MCANDIV = 0000<sub>B</sub> the clock is shut off.</p> <ul style="list-style-type: none"> <li>0<sub>H</sub> <math>f_{\text{MCANI}}</math> is stopped</li> <li>1<sub>H</sub> <math>f_{\text{MCANI}} = f_{\text{source1}}</math></li> <li>2<sub>H</sub> <math>f_{\text{MCANI}} = f_{\text{source1}}/2</math></li> <li>3<sub>H</sub> <math>f_{\text{MCANI}} = f_{\text{source1}}/3</math></li> <li>4<sub>H</sub> <math>f_{\text{MCANI}} = f_{\text{source1}}/4</math></li> <li>5<sub>H</sub> <math>f_{\text{MCANI}} = f_{\text{source1}}/5</math></li> <li>6<sub>H</sub> <math>f_{\text{MCANI}} = f_{\text{source1}}/6</math></li> <li>7<sub>H</sub> Reserved, do not use this combination</li> <li>8<sub>H</sub> <math>f_{\text{MCANI}} = f_{\text{source1}}/8</math></li> <li>9<sub>H</sub> Reserved, do not use this combination</li> <li>A<sub>H</sub> <math>f_{\text{MCANI}} = f_{\text{source1}}/10</math></li> <li>B<sub>H</sub> Reserved, do not use this combination</li> <li>C<sub>H</sub> <math>f_{\text{MCANI}} = f_{\text{source1}}/12</math></li> <li>D<sub>H</sub> Reserved, do not use this combination</li> <li>E<sub>H</sub> Reserved, do not use this combination</li> <li>F<sub>H</sub> <math>f_{\text{MCANI}} = f_{\text{source1}}/15</math></li> </ul>
<b>CLKSELMCAN</b>	5:4	rw	<p><b>Clock Selection for MCAN</b></p> <p>This bit field defines the clock source that is used for the clock generation of <math>f_{\text{SOURCEMCAN}}</math>.</p> <p><b>Note:</b> For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to 00<sub>B</sub>. Second step is to switch to the new target configuration.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> <math>f_{\text{MCAN}}</math> clock is stopped</li> <li>01<sub>B</sub> <math>f_{\text{MCANI}}</math> is used as clock source <math>f_{\text{MCAN}}</math></li> <li>10<sub>B</sub> <math>f_{\text{osco}}</math> is used as clock source <math>f_{\text{MCAN}}</math></li> <li>11<sub>B</sub> Reserved, do not use this combination</li> </ul>
<b>PLL1DIVDIS</b>	7	rw	<p><b>Divider Disable for fPLL1</b></p> <p>Depending on CCUCON0.CLKSEL, this bit selects whether <math>f_{\text{source1}}</math> is half <math>f_{\text{pll1}}</math>.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> CLKSEL != 01 <math>\rightarrow f_{\text{source1}} = f_{\text{back}}</math> CLKSEL = 01 <math>\rightarrow f_{\text{source1}} = f_{\text{pll1}}/2</math></li> <li>1<sub>B</sub> CLKSEL != 01 <math>\rightarrow f_{\text{source1}} = f_{\text{back}}</math> CLKSEL = 01 <math>\rightarrow f_{\text{source1}} = f_{\text{pll1}}</math></li> </ul>

## Clocking System

Field	Bits	Type	Description
I2CDIV	11:8	rw	<p><b>I2C Divider Reload Value</b></p> <p>The resulting I2C frequency is configured to <math>f_{I2C} = f_{SOURCE2} / I2CDIV</math> for the allowed configurations. For <math>I2CDIV = 0000_B</math> the clock is shut off.</p> <ul style="list-style-type: none"> <li>0<sub>H</sub> <math>f_{I2C}</math> is stopped</li> <li>1<sub>H</sub> <math>f_{I2C} = f_{SOURCE2}</math></li> <li>2<sub>H</sub> <math>f_{I2C} = f_{SOURCE2}/2</math></li> <li>3<sub>H</sub> <math>f_{I2C} = f_{SOURCE2}/3</math></li> <li>4<sub>H</sub> <math>f_{I2C} = f_{SOURCE2}/4</math></li> <li>5<sub>H</sub> <math>f_{I2C} = f_{SOURCE2}/5</math></li> <li>6<sub>H</sub> <math>f_{I2C} = f_{SOURCE2}/6</math></li> <li>7<sub>H</sub> Reserved, do not use this combination</li> <li>8<sub>H</sub> <math>f_{I2C} = f_{SOURCE2}/8</math></li> <li>9<sub>H</sub> Reserved, do not use this combination</li> <li>A<sub>H</sub> <math>f_{I2C} = f_{SOURCE2}/10</math></li> <li>B<sub>H</sub> Reserved, do not use this combination</li> <li>C<sub>H</sub> <math>f_{I2C} = f_{SOURCE2}/12</math></li> <li>D<sub>H</sub> Reserved, do not use this combination</li> <li>E<sub>H</sub> Reserved, do not use this combination</li> <li>F<sub>H</sub> <math>f_{I2C} = f_{SOURCE2}/15</math></li> </ul>
MSCDIV	19:16	rw	<p><b>MSC Divider Reload Value</b></p> <p>The resulting MSC frequency is configured to <math>f_{MSC} = f_{SOURCEMSC} / MSCDIV</math> for the allowed configurations. For <math>MSCDIV = 0000_B</math> the clock is shut off.</p> <ul style="list-style-type: none"> <li>0<sub>H</sub> <math>f_{MSC}</math> is stopped</li> <li>1<sub>H</sub> <math>f_{MSC} = f_{SOURCEMSC}</math></li> <li>2<sub>H</sub> <math>f_{MSC} = f_{SOURCEMSC}/2</math></li> <li>3<sub>H</sub> <math>f_{MSC} = f_{SOURCEMSC}/3</math></li> <li>4<sub>H</sub> <math>f_{MSC} = f_{SOURCEMSC}/4</math></li> <li>5<sub>H</sub> <math>f_{MSC} = f_{SOURCEMSC}/5</math></li> <li>6<sub>H</sub> <math>f_{MSC} = f_{SOURCEMSC}/6</math></li> <li>7<sub>H</sub> Reserved, do not use this combination</li> <li>8<sub>H</sub> <math>f_{MSC} = f_{SOURCEMSC}/8</math></li> <li>9<sub>H</sub> Reserved, do not use this combination</li> <li>A<sub>H</sub> <math>f_{MSC} = f_{SOURCEMSC}/10</math></li> <li>B<sub>H</sub> Reserved, do not use this combination</li> <li>C<sub>H</sub> <math>f_{MSC} = f_{SOURCEMSC}/12</math></li> <li>D<sub>H</sub> Reserved, do not use this combination</li> <li>E<sub>H</sub> Reserved, do not use this combination</li> <li>F<sub>H</sub> <math>f_{MSC} = f_{SOURCEMSC}/15</math></li> </ul>

## Clocking System

Field	Bits	Type	Description
CLKSELMSC	21:20	rw	<p><b>Clock Selection for MSC</b></p> <p>This bit field defines the clock source that is used for the clock generation of <math>f_{\text{SOURCEMSC}}</math>.</p> <p><b>Note:</b> For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to <math>00_B</math>. Second step is to switch to the new target configuration.</p> <p> <math>00_B</math> <math>f_{\text{MSC}}</math> clock is stopped  <math>01_B</math> <math>f_{\text{source1}}</math> is used as clock source <math>f_{\text{SOURCEMSC}}</math>  <math>10_B</math> <math>f_{\text{source2}}</math> is used as clock source <math>f_{\text{SOURCEMSC}}</math>  <math>11_B</math> Reserved, do not use this combination     </p>
QSPIDIV	27:24	rw	<p><b>QSPI Divider Reload Value</b></p> <p>The resulting QSPI frequency is configured to <math>f_{\text{QSPI}} = f_{\text{SOURCEQSPI}} / \text{QSPIDIV}</math> for the allowed configurations. For <math>\text{QSPIDIV} = 0000_B</math> the clock is shut off.</p> <p> <math>0_H</math> <math>f_{\text{QSPI}}</math> is stopped  <math>1_H</math> <math>f_{\text{QSPI}} = f_{\text{SOURCEQSPI}}</math>  <math>2_H</math> <math>f_{\text{QSPI}} = f_{\text{SOURCEQSPI}}/2</math>  <math>3_H</math> <math>f_{\text{QSPI}} = f_{\text{SOURCEQSPI}}/3</math>  <math>4_H</math> <math>f_{\text{QSPI}} = f_{\text{SOURCEQSPI}}/4</math>  <math>5_H</math> <math>f_{\text{QSPI}} = f_{\text{SOURCEQSPI}}/5</math>  <math>6_H</math> <math>f_{\text{QSPI}} = f_{\text{SOURCEQSPI}}/6</math>  <math>7_H</math> Reserved, do not use this combination  <math>8_H</math> <math>f_{\text{QSPI}} = f_{\text{SOURCEQSPI}}/8</math>  <math>9_H</math> Reserved, do not use this combination  <math>A_H</math> <math>f_{\text{QSPI}} = f_{\text{SOURCEQSPI}}/10</math>  <math>B_H</math> Reserved, do not use this combination  <math>C_H</math> <math>f_{\text{QSPI}} = f_{\text{SOURCEQSPI}}/12</math>  <math>D_H</math> Reserved, do not use this combination  <math>E_H</math> Reserved, do not use this combination  <math>F_H</math> <math>f_{\text{QSPI}} = f_{\text{SOURCEQSPI}}/15</math> </p>
CLKSELQSPI	29:28	rw	<p><b>Clock Selection for QSPI</b></p> <p>This bit field defines the clock source that is used for the clock generation of <math>f_{\text{SOURCEQSPI}}</math>.</p> <p><b>Note:</b> For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to <math>00_B</math>. Second step is to switch to the new target configuration.</p> <p> <math>00_B</math> <math>f_{\text{QSPI}}</math> clock is stopped  <math>01_B</math> <math>f_{\text{source1}}</math> is used as clock source <math>f_{\text{SOURCEQSPI}}</math>  <math>10_B</math> <math>f_{\text{source2}}</math> is used as clock source <math>f_{\text{SOURCEQSPI}}</math>  <math>11_B</math> Reserved, do not use this combination     </p>

## Clocking System

Field	Bits	Type	Description
LCK	31	rh	<p><b>Lock Status</b>            This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <p><i>Note:</i> <i>The lock bit is set when at least one bit field is changed, and released when this change is executed.</i></p> <p>0<sub>B</sub> The register is unlocked and can be updated            1<sub>B</sub> The register is locked and can not be updated</p>
0	6, 15:12, 23:22, 30	rw	<p><b>Reserved</b>            Should be written with 0.</p>

### CCU Clock Control Register 2

CCUCON2

CCU Clock Control Register 2

(0040<sub>H</sub>)System Reset Value: 0700 0101<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK		0		HSPD MPER ON	ERAYP ERON	EBUPE RON					0				
rh		rw		rw	rw	rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CLKSELASCLI NS			ASCLINSDIV			0				0		ASCLINFDIV		

## Clocking System

Field	Bits	Type	Description
<b>ASCLINFDIV</b>	3:0	rw	<p><b>ASCLIN Fast Divider Reload Value</b></p> <p>The resulting ASCLIN frequency is configured to <math>f_{ASCLINF} = f_{source2} / ASCLINFDIV</math> for the allowed configurations. For <math>ASCLINFDIV = 0000_B</math> the clock is shut off. <math>f_{source2}</math> could be configured either to <math>f_{PLL2}</math> (<math>CLKSEL = 01_B</math>) or <math>f_{BACK}</math> (<math>CLKSEL = 00_B</math>)</p> <p>0<sub>H</sub> <math>f_{ASCLINF}</math> is stopped      1<sub>H</sub> <math>f_{ASCLINF} = f_{source2}</math>      2<sub>H</sub> <math>f_{ASCLINF} = f_{source2}/2</math>      3<sub>H</sub> <math>f_{ASCLINF} = f_{source2}/3</math>      4<sub>H</sub> <math>f_{ASCLINF} = f_{source2}/4</math>      5<sub>H</sub> <math>f_{ASCLINF} = f_{source2}/5</math>      6<sub>H</sub> <math>f_{ASCLINF} = f_{source2}/6</math>      7<sub>H</sub> Reserved, do not use this combination      8<sub>H</sub> <math>f_{ASCLINF} = f_{source2}/8</math>      9<sub>H</sub> Reserved, do not use this combination      A<sub>H</sub> <math>f_{ASCLINF} = f_{source2}/10</math>      B<sub>H</sub> Reserved, do not use this combination      C<sub>H</sub> <math>f_{ASCLINF} = f_{source2}/12</math>      D<sub>H</sub> Reserved, do not use this combination      E<sub>H</sub> Reserved, do not use this combination      F<sub>H</sub> <math>f_{ASCLINF} = f_{source2}/15</math></p>
<b>ASCLINSDIV</b>	11:8	rw	<p><b>ASCLIN Slow Divider Reload Value</b></p> <p>The resulting ASCLIN frequency is configured to <math>f_{ASCLINSI} = f_{source1} / ASCLINSDIV</math> for the allowed configurations. For <math>ASCLINSDIV = 0000_B</math> the clock is shut off. <math>f_{source1}</math> could be configured either to <math>f_{PLL1}</math> (<math>CLKSEL = 01_B</math>) or <math>f_{BACK}</math> (<math>CLKSEL = 00_B</math>)</p> <p>0<sub>H</sub> <math>f_{ASCLINSI}</math> is stopped      1<sub>H</sub> <math>f_{ASCLINSI} = f_{source1}</math>      2<sub>H</sub> <math>f_{ASCLINSI} = f_{source1}/2</math>      3<sub>H</sub> <math>f_{ASCLINSI} = f_{source1}/3</math>      4<sub>H</sub> <math>f_{ASCLINSI} = f_{source1}/4</math>      5<sub>H</sub> <math>f_{ASCLINSI} = f_{source1}/5</math>      6<sub>H</sub> <math>f_{ASCLINSI} = f_{source1}/6</math>      7<sub>H</sub> Reserved, do not use this combination      8<sub>H</sub> <math>f_{ASCLINSI} = f_{source1}/8</math>      9<sub>H</sub> Reserved, do not use this combination      A<sub>H</sub> <math>f_{ASCLINSI} = f_{source1}/10</math>      B<sub>H</sub> Reserved, do not use this combination      C<sub>H</sub> <math>f_{ASCLINSI} = f_{source1}/12</math>      D<sub>H</sub> Reserved, do not use this combination      E<sub>H</sub> Reserved, do not use this combination      F<sub>H</sub> <math>f_{ASCLINSI} = f_{source1}/15</math></p>

## Clocking System

Field	Bits	Type	Description
<b>CLKSELASCLIN S</b>	13:12	rw	<p><b>Clock Selection for ASCLINS</b></p> <p>This bit field defines the clock source that is used for the clock generation of <math>f_{ASCLINS}</math>.</p> <p><i>Note:</i> For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to <math>00_B</math>. Second step is to switch to the new target configuration.</p> <p> <math>00_B</math> <math>f_{ASCLINS}</math> clock is stopped  <math>01_B</math> <math>f_{ASCLINSI}</math> is used as clock <math>f_{ASCLINS}</math>  <math>10_B</math> <math>f_{OSCO}</math> is used as clock <math>f_{ASCLINS}</math>  <math>11_B</math> Reserved, do not use this combination     </p>
<b>EBUPERON</b>	24	rw	<p><b>Power Safe SwitchOff for EBU Clock</b></p> <p>This bit is used to control the EBU peripheral clock <math>f_{EBU}</math> for power saving purposes if the logic is not used by the application.</p> <p> <math>0_B</math> <math>f_{EBU}</math> is stopped  <math>1_B</math> <math>f_{EBU} = f_{source1}</math> </p>
<b>ERAYPERON</b>	25	rw	<p><b>Power Safe SwitchOff for ERAY Clock</b></p> <p>This bit is used to control the ERAY peripheral clock <math>f_{ERAY}</math> for power saving purposes if the logic is not used by the application.</p> <p> <math>0_B</math> <math>f_{ERAY}</math> is stopped  <math>1_B</math> <math>f_{ERAY} = f_{source1} / 2</math> </p>
<b>HSPDMPERON</b>	26	rw	<p><b>Power Safe SwitchOff for HSPDM Clocks</b></p> <p>This bit is used to control the HSPDM peripheral clocks <math>f_{HSPDM\_320}</math> and <math>f_{HSPDM\_160}</math> for power saving purposes if the logic is not used by the application.</p> <p> <math>0_B</math> <math>f_{HSPDM\_320}</math> is stopped; <math>f_{HSPDM\_160}</math> is stopped  <math>1_B</math> <math>f_{HSPDM\_320} = f_{src1}; f_{HSPDM\_160} = f_{source1}</math> </p>
<b>LCK</b>	31	rh	<p><b>Lock Status</b></p> <p>This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <p><i>Note:</i> The lock bit is set when at least one bit field is changed, and released when this change is executed.</p> <p> <math>0_B</math> The register is unlocked and can be updated  <math>1_B</math> The register is locked and can not be updated     </p>
<b>0</b>	7:4, 23:14, 30:27	rw	<p><b>Reserved</b></p> <p>Should be written with 0.</p>

## Clocking System

### CCU Clock Control Register 5

CCUCON5

CCU Clock Control Register 5

(004C<sub>H</sub>)System Reset Value: 0000 0030<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	<b>UP</b>								<b>0</b>						
rh	w								rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				<b>0</b>				<b>ADASDIV</b>			<b>MCANHDIV</b>		<b>GETHDIV</b>		
				rw				rw			rw		rw		

Field	Bits	Type	Description																																
<b>GETHDIV</b>	3:0	rw	<p><b>GETH Divider Reload Value</b></p> <p>The resulting GETH frequency is configured to <math>f_{GETH} = f_{source0} / \text{GETHDIV}</math> for the allowed configurations. For <math>\text{GETHDIV} = 0000_B</math> the clock is shut off. <math>f_{source0}</math> could be configured either to <math>f_{PLL0}</math> (<math>\text{CLKSEL} = 01_B</math>) or <math>f_{BACK}</math> (<math>\text{CLKSEL} = 00_B</math>)</p> <p><b>Note:</b> <i>GETHDIV must be enabled (!=0) during an application reset to allow firmware related installation tasks.</i></p> <table> <tr> <td><math>0_H</math></td><td><math>f_{GETH}</math> is stopped</td></tr> <tr> <td><math>1_H</math></td><td><math>f_{GETH} = f_{source0}</math></td></tr> <tr> <td><math>2_H</math></td><td><math>f_{GETH} = f_{source0}/2</math></td></tr> <tr> <td><math>3_H</math></td><td><math>f_{GETH} = f_{source0}/3</math></td></tr> <tr> <td><math>4_H</math></td><td><math>f_{GETH} = f_{source0}/4</math></td></tr> <tr> <td><math>5_H</math></td><td>Reserved, do not use this combination</td></tr> <tr> <td><math>6_H</math></td><td>Reserved, do not use this combination</td></tr> <tr> <td><math>7_H</math></td><td>Reserved, do not use this combination</td></tr> <tr> <td><math>8_H</math></td><td>Reserved, do not use this combination</td></tr> <tr> <td><math>9_H</math></td><td>Reserved, do not use this combination</td></tr> <tr> <td><math>A_H</math></td><td>Reserved, do not use this combination</td></tr> <tr> <td><math>B_H</math></td><td>Reserved, do not use this combination</td></tr> <tr> <td><math>C_H</math></td><td>Reserved, do not use this combination</td></tr> <tr> <td>...</td><td></td></tr> <tr> <td><math>E_H</math></td><td>Reserved, do not use this combination</td></tr> <tr> <td><math>F_H</math></td><td>Reserved, do not use this combination</td></tr> </table>	$0_H$	$f_{GETH}$ is stopped	$1_H$	$f_{GETH} = f_{source0}$	$2_H$	$f_{GETH} = f_{source0}/2$	$3_H$	$f_{GETH} = f_{source0}/3$	$4_H$	$f_{GETH} = f_{source0}/4$	$5_H$	Reserved, do not use this combination	$6_H$	Reserved, do not use this combination	$7_H$	Reserved, do not use this combination	$8_H$	Reserved, do not use this combination	$9_H$	Reserved, do not use this combination	$A_H$	Reserved, do not use this combination	$B_H$	Reserved, do not use this combination	$C_H$	Reserved, do not use this combination	...		$E_H$	Reserved, do not use this combination	$F_H$	Reserved, do not use this combination
$0_H$	$f_{GETH}$ is stopped																																		
$1_H$	$f_{GETH} = f_{source0}$																																		
$2_H$	$f_{GETH} = f_{source0}/2$																																		
$3_H$	$f_{GETH} = f_{source0}/3$																																		
$4_H$	$f_{GETH} = f_{source0}/4$																																		
$5_H$	Reserved, do not use this combination																																		
$6_H$	Reserved, do not use this combination																																		
$7_H$	Reserved, do not use this combination																																		
$8_H$	Reserved, do not use this combination																																		
$9_H$	Reserved, do not use this combination																																		
$A_H$	Reserved, do not use this combination																																		
$B_H$	Reserved, do not use this combination																																		
$C_H$	Reserved, do not use this combination																																		
...																																			
$E_H$	Reserved, do not use this combination																																		
$F_H$	Reserved, do not use this combination																																		

## Clocking System

Field	Bits	Type	Description
<b>MCANHDIV</b>	7:4	rw	<p><b>MCANH Divider Reload Value</b></p> <p>The resulting MCANH frequency is configured to <math>f_{\text{MCANH}} = f_{\text{SOURCE0}} / \text{MCANHDIV}</math> for the allowed configurations. For MCANHDIV = 0000<sub>B</sub> the clock is shut off.</p> <ul style="list-style-type: none"> <li>0<sub>H</sub> <math>f_{\text{MCANH}}</math> is stopped</li> <li>1<sub>H</sub> <math>f_{\text{MCANH}} = f_{\text{SOURCE0}}</math></li> <li>2<sub>H</sub> <math>f_{\text{MCANH}} = f_{\text{SOURCE0}}/2</math></li> <li>3<sub>H</sub> <math>f_{\text{MCANH}} = f_{\text{SOURCE0}}/3</math></li> <li>4<sub>H</sub> <math>f_{\text{MCANH}} = f_{\text{SOURCE0}}/4</math></li> <li>5<sub>H</sub> <math>f_{\text{MCANH}} = f_{\text{SOURCE0}}/5</math></li> <li>6<sub>H</sub> <math>f_{\text{MCANH}} = f_{\text{SOURCE0}}/6</math></li> <li>7<sub>H</sub> Reserved, do not use this combination</li> <li>8<sub>H</sub> <math>f_{\text{MCANH}} = f_{\text{SOURCE0}}/8</math></li> <li>9<sub>H</sub> Reserved, do not use this combination</li> <li>A<sub>H</sub> <math>f_{\text{MCANH}} = f_{\text{SOURCE0}}/10</math></li> <li>B<sub>H</sub> Reserved, do not use this combination</li> <li>C<sub>H</sub> <math>f_{\text{MCANH}} = f_{\text{SOURCE0}}/12</math></li> <li>D<sub>H</sub> Reserved, do not use this combination</li> <li>E<sub>H</sub> Reserved, do not use this combination</li> <li>F<sub>H</sub> <math>f_{\text{MCANH}} = f_{\text{SOURCE0}}/15</math></li> </ul>
<b>ADASDIV</b>	11:8	rw	<p><b>ADAS Divider Reload Value</b></p> <p>The resulting ADAS frequency is configured to <math>f_{\text{ADAS}} = f_{\text{source0}} / \text{ADASDIV}</math> for the allowed configurations. <math>f_{\text{source0}}</math> could be configured either to <math>f_{\text{PLL0}}</math> (CLKSEL = 01<sub>B</sub>) or <math>f_{\text{BACK}}</math> (CLKSEL = 00<sub>B</sub>)</p> <p><i>Note:</i> ADASDIV must be enabled (!=0) during an application reset to allow firmware related installation tasks.</p> <ul style="list-style-type: none"> <li>0<sub>H</sub> <math>f_{\text{ADAS}}</math> is stopped</li> <li>1<sub>H</sub> <math>f_{\text{ADAS}} = f_{\text{source0}}</math></li> <li>2<sub>H</sub> <math>f_{\text{ADAS}} = f_{\text{SOURCE0}}/2</math></li> <li>3<sub>H</sub> <math>f_{\text{ADAS}} = f_{\text{SOURCE0}}/3</math></li> <li>4<sub>H</sub> <math>f_{\text{ADAS}} = f_{\text{SOURCE0}}/4</math></li> <li>5<sub>H</sub> <math>f_{\text{ADAS}} = f_{\text{SOURCE0}}/5</math></li> <li>6<sub>H</sub> <math>f_{\text{ADAS}} = f_{\text{SOURCE0}}/6</math></li> <li>7<sub>H</sub> Reserved, do not use this combination;</li> <li>8<sub>H</sub> <math>f_{\text{ADAS}} = f_{\text{SOURCE0}}/8</math></li> <li>9<sub>H</sub> Reserved, do not use this combination</li> <li>A<sub>H</sub> <math>f_{\text{ADAS}} = f_{\text{SOURCE0}}/10</math></li> <li>B<sub>H</sub> Reserved, do not use this combination</li> <li>C<sub>H</sub> <math>f_{\text{ADAS}} = f_{\text{SOURCE0}}/12</math></li> <li>D<sub>H</sub> Reserved, do not use this combination</li> <li>E<sub>H</sub> <math>f_{\text{ADAS}} = f_{\text{SOURCE0}}/12</math></li> <li>F<sub>H</sub> <math>f_{\text{ADAS}} = f_{\text{SOURCE0}}/15</math></li> </ul>

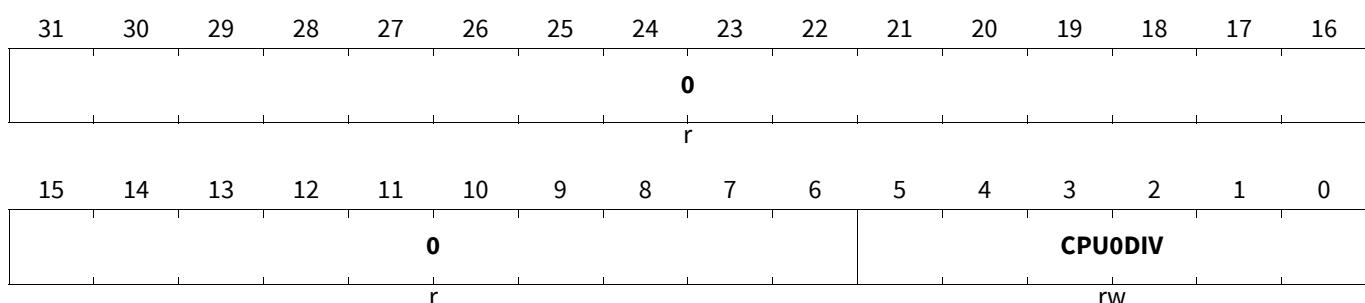
## Clocking System

Field	Bits	Type	Description
UP	30	w	<p><b>Update Request</b>  Setting this bit will request an update for CCUCON0 and CCUCON5. Only one UP bit must be set either CCUCON0 or CCUCON5. This bit always reads as zero.</p> <p>0<sub>B</sub> No action  1<sub>B</sub> A new complete parameter set is transferred to the CCU defined by register CCUCON0 and CCUCON5.</p>
LCK	31	rh	<p><b>Lock Status</b>  This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <p><i>Note:</i> The lock bit is set when an update of CCUCON0/5 has been requested, and released when the update is complete.</p> <p>0<sub>B</sub> The register is unlocked and can be updated  1<sub>B</sub> The register is locked and can not be updated</p>
0	29:12	rw	<p><b>Reserved</b>  Should be written with 0.</p>

### CCU Clock Control Register 6

#### CCUCON6

#### CCU Clock Control Register 6

(0080<sub>H</sub>)System Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
CPU0DIV	5:0	rw	<p><b>CPU0 Divider Reload Value</b>  The resulting CPU0 frequency (performance) is configured to <math>f_{CPU0} = f_{SRI} * (64 - CPU0DIV) / 64</math>. For CPU0DIV = 000000<sub>B</sub>, <math>f_{CPU0} = f_{SRI}</math>.</p>
0	31:6	r	<p><b>Reserved</b>  Read as 0; Should be written with 0.</p>

## Clocking System

### CCU Clock Control Register 7

CCUCON7

CCU Clock Control Register 7

(0084<sub>H</sub>)System Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								CPU1DIV							
r															

Field	Bits	Type	Description
CPU1DIV	5:0	rw	<b>CPU1 Divider Reload Value</b> The resulting CPU1 frequency (performance) is configured to $f_{CPU1} = f_{SRI} * (64 - CPU1DIV) / 64$ . For CPU1DIV = 000000 <sub>B</sub> , $f_{CPU1} = f_{SRI}$ .
0	31:6	r	<b>Reserved</b> Read as 0; Should be written with 0.

### CCU Clock Control Register 8

CCUCON8

CCU Clock Control Register 8

(0088<sub>H</sub>)System Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								CPU2DIV							
r															

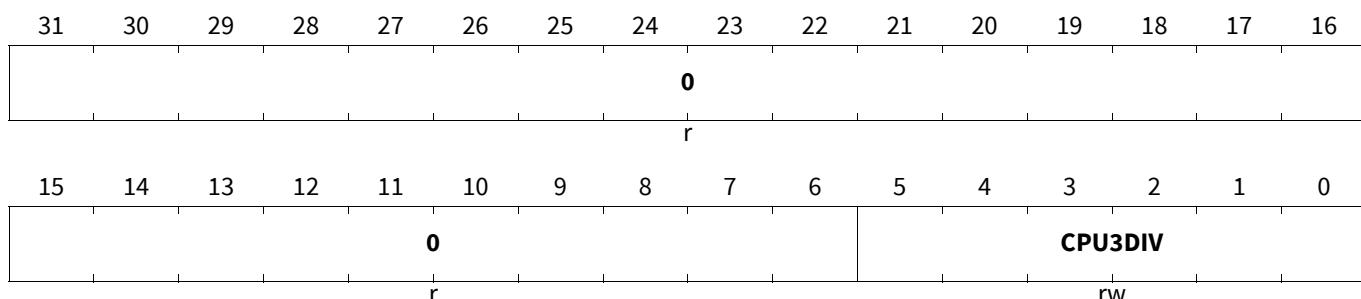
Field	Bits	Type	Description
CPU2DIV	5:0	rw	<b>CPU2 Divider Reload Value</b> The resulting CPU2 frequency (performance) is configured to $f_{CPU2} = f_{SRI} * (64 - CPU2DIV) / 64$ . For CPU2DIV = 000000 <sub>B</sub> , $f_{CPU2} = f_{SRI}$ .
0	31:6	r	<b>Reserved</b> Read as 0; Should be written with 0.

## Clocking System

### CCU Clock Control Register 9

CCUCON9

CCU Clock Control Register 9

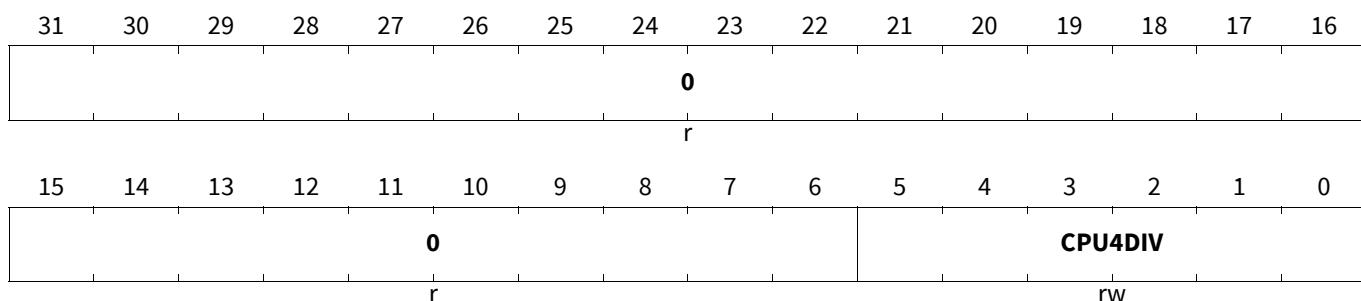
(008C<sub>H</sub>)System Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
CPU3DIV	5:0	rw	<b>CPU3 Divider Reload Value</b> The resulting CPU3 frequency (performance) is configured to $f_{CPU3} = f_{SRI} * (64 - CPU3DIV) / 64$ . For CPU3DIV = 000000 <sub>B</sub> , $f_{CPU3} = f_{SRI}$ .
0	31:6	r	<b>Reserved</b> Read as 0; Should be written with 0.

### CCU Clock Control Register 10

CCUCON10

CCU Clock Control Register 10

(0090<sub>H</sub>)System Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
CPU4DIV	5:0	rw	<b>CPU4 Divider Reload Value</b> The resulting CPU4 frequency (performance) is configured to $f_{CPU4} = f_{SRI} * (64 - CPU4DIV) / 64$ . For CPU4DIV = 000000 <sub>B</sub> , $f_{CPU4} = f_{SRI}$ .
0	31:6	r	<b>Reserved</b> Read as 0; Should be written with 0.

## Clocking System

### CCU Clock Control Register 11

#### CCUCON11

#### CCU Clock Control Register 11

(0094<sub>H</sub>)

System Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										CPU5DIV					
r															

Field	Bits	Type	Description
CPU5DIV	5:0	rw	<b>CPU5 Divider Reload Value</b> The resulting CPU5 frequency (performance) is configured to $f_{CPU5} = f_{SRI} * (64 - CPU5DIV) / 64$ . For CPU5DIV = 000000 <sub>B</sub> , $f_{CPU5} = f_{SRI}$ .
0	31:6	r	<b>Reserved</b> Read as 0; Should be written with 0.

## 10.6 Clock Emergency Behavior

In case of a clock error, the CCU switches to the back-up clock  $f_{BACK}$  as the clock source.

A clock error is defined by the occurrence of at least one of the following conditions:

- loss of lock event of the System PLL while selected as clock source for the CCU (CLKSEL = 01<sub>B</sub>)
- loss of lock event of the Peripheral PLL while selected as clock source for the CCU (CLKSEL = 01<sub>B</sub>)

A clock error, as described above, is active until the next time when CCUCON0.UP is set to transfer a new CLKSEL value (even if it is the old one).

A clock error will also trigger an SMU alarm.

**Note:** After a clock emergency, the bit fields CLKSELx have to be re-written. After the root cause for the clock error disappears, the application can reconfigure the clock system, including CLKSELx.

## 10.7 External Clock Output

Two external clock outputs are provided via pins EXTCLK0 and EXTCLK1. These external clocks can be enabled/disabled via bits EXTCON.EN0 for EXTCLK0 and EXTCON.EN1 for EXTCLK1. Each of the clocks that defines a clock domain can individually be selected to be seen at pins EXTCLK0 or EXTCLK1; this is configured via bit field EXTCON.SEL0/1. Changing the content of bit field EXTCON.SEL0/1 can lead to spikes at pins EXTCLK0/1.

### 10.7.1 Programmable Frequency Output for EXTCLK0

This section describes external clock generation using the Fractional Divider.

#### Overview

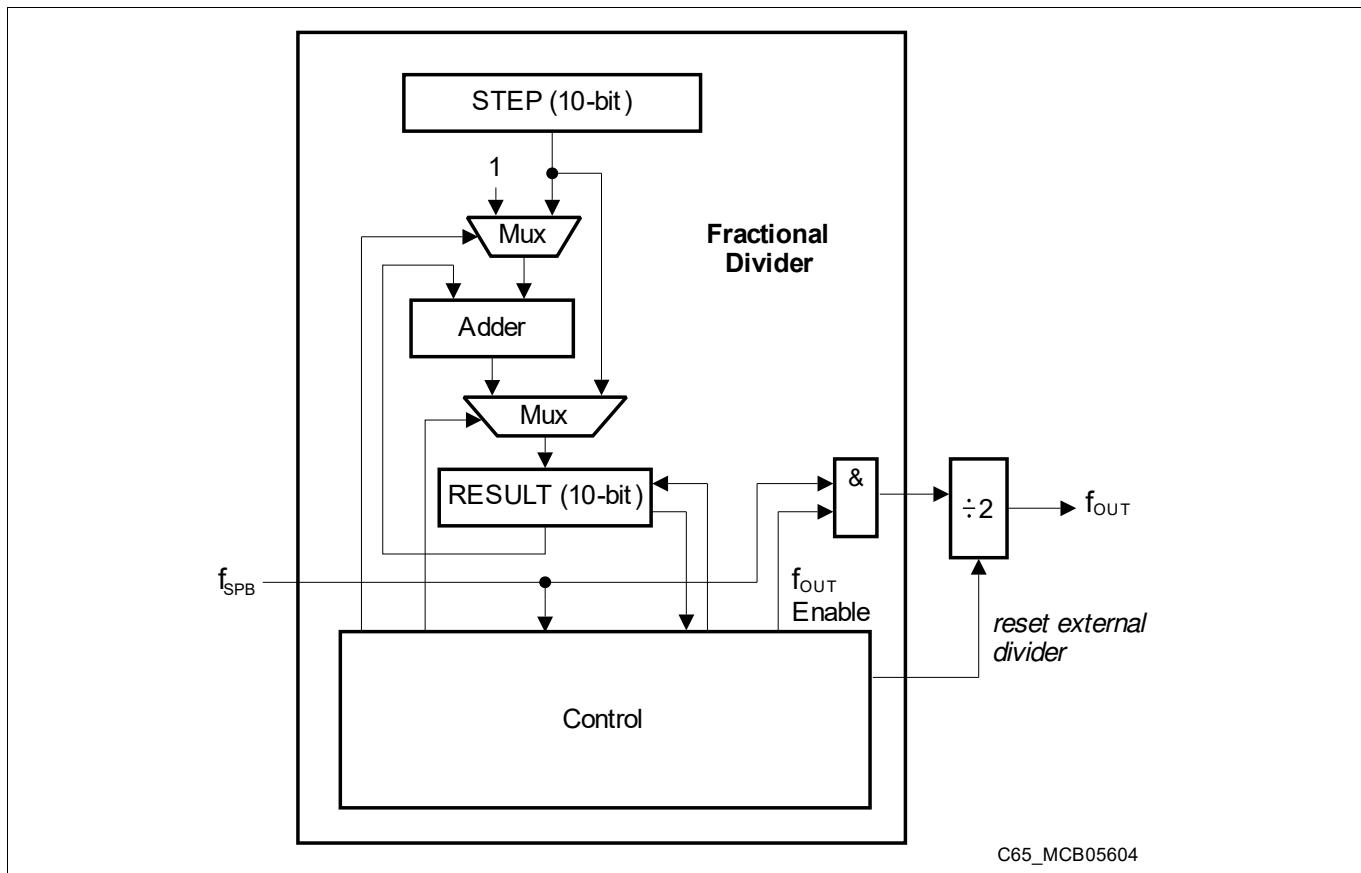
The fractional divider makes it possible to generate an external clock from the SPB clock using a programmable divider. The fractional divider divides the input clock  $f_{SPB}$  either by the factor 1/n or by a fraction of n/1024 for any

## Clocking System

value of n from 0 to 1023. This clock is thereafter divided additionally by a factor of two to guarantee a 50% duty cycle and outputs the clock  $f_{OUT}$ . The fractional divider is controlled by the FDR register. **Figure 88** shows the fractional divider block diagram.

The adder logic of the fractional divider can be configured for two operating modes:

- Reload counter (addition of +1), generating an output clock pulse on counter overflow
- Adder that adds a STEP value to the RESULT value and generates an output clock pulse on counter overflow



**Figure 88 Fractional Divider Block Diagram**

The adder logic of the fractional divider can be configured for two operating modes:

- **Normal Mode**: Reload counter ( $RESULT = RESULT + 1$ ), generating an output clock pulse on counter overflow
- **Fractional Divider Mode**: Adder that adds a STEP value to the RESULT value and generates an output clock pulse on counter overflow

### 10.7.1.1 Fractional Divider Operating Modes

The fractional divider has two operating modes:

- Normal Divider Mode
- Fractional Divider Mode

#### Normal Divider Mode

In Normal Divider Mode (FDR.DM = 01<sub>B</sub>), the fractional divider behaves as a reload counter (addition of +1) that generates an output clock pulse on the transition from 3FF<sub>H</sub> to 000<sub>H</sub>. FDR.RESULT represents the counter value and FDR.STEP determines the reload value.

## Clocking System

The output frequencies in Normal Divider Mode are defined according to the following formulas:

$$f_{\text{OUT}} = \frac{f_{\text{SPB}} \times \frac{1}{n}}{2}, \text{ with } n = 1024 - \text{STEP} \quad (10.16)$$

In order to get  $f_{\text{OUT}} = f_{\text{SPB}}/2$  STEP must be programmed with  $3FF_H$ .

### Fractional Divider Mode

When the Fractional Divider Mode is selected ( $\text{FDR.DM} = 10_B$ ), the output is derived from the input clock  $f_{\text{SPB}}$  by division of a fraction of  $n/1024$ , for any value of  $n$  from 0 to 1023, followed by the division of two. In general, the Fractional Divider Mode makes it possible to program the average output clock frequency with a higher accuracy than in Normal Divider Mode.

In Fractional Divider Mode, a pulse is generated depending on the result of the addition  $\text{FDR.RESULT} + \text{FDR.STEP}$ . If the addition leads to an overflow over  $3FF_H$ , a pulse is generated for the divider by two. Note that in Fractional Divider Mode, the clock  $f_{\text{OUT}}$  can have a maximum period jitter of one  $f_{\text{SPB}}$  clock period.

The output frequencies in Fractional Divider Mode are defined according to the following formulas:

$$f_{\text{OUT}} = \frac{f_{\text{SPB}} \times \frac{\text{STEP}}{1024}}{2} \quad (10.17)$$

### 10.7.2 Programmable Frequency Output for EXTCLK1

Clock  $f_{\text{OUT}}$  is generated via a counter, so the output frequency can be selected in small steps.

$f_{\text{OUT}}$  always provides complete output periods.

Register EXTCON provides control over the output generation (frequency and activation).

**Clocking System****10.7.3 Clock Output Control Register**

Clock Output Control registers can be accessed by all CPUs in the system. However, it is suggested that only one CPU is used to control the clocks. As CPU0 is the active and available CPU after each reset, this is the best choice.

**External Clock Control Register****EXTCON****External Clock Control Register****(003C<sub>H</sub>)****System Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>DIV1</b>															
								0							
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															
SEL0															
rwh															
0															
rwh															

Field	Bits	Type	Description
<b>ENO</b>	0	rwh	<b>External Clock Enable for EXTCLK0</b>  <i>Note:</i> If the generation of the external clock signal is disabled, the signal is tied to zero.  $0_B$ No external clock is provided $1_B$ The configured external clock is provided
<b>SEL0</b>	5:2	rwh	<b>External Clock Select for EXTCLK0</b> This bit field defines the clock source that is selected as output for pin EXTCLK0. $0_H$ $f_{OUT}$ is selected for the external clock signal $1_H$ $f_{PLL0}$ is selected for the external clock signal $2_H$ $f_{PLL1}$ is selected for the external clock signal $3_H$ $f_{OSCO}$ is selected for the external clock signal $4_H$ $f_{BACK}$ is selected for the external clock signal $5_H$ $f_{PLL2}$ is selected for the external clock signal $6_H$ $f_{BBB}$ is selected for the external clock signal $7_H$ Reserved, do not use this configuration $8_H$ $f_{SRI}$ is selected for the external clock signal $9_H$ $f_{SPB}$ is selected for the external clock signal $A_H$ $f_{FSI}$ is selected for the external clock signal $B_H$ $f_{STM}$ is selected for the external clock signal $C_H$ $f_{GTM}$ is selected for the external clock signal $D_H$ $f_{TCK}$ is selected for the external clock signal $E_H$ $f_{FSI2}$ is selected for the external clock signal $F_H$ $f_{MTO}$ from the ERAY module is selected for the external clock signal

## Clocking System

Field	Bits	Type	Description
<b>EN1</b>	16	rwh	<p><b>External Clock Enable for EXTCLK1</b></p> <p><b>Note:</b> If the generation of the external clock signal is disabled, the signal is tied to zero.</p> <p>0<sub>B</sub> No external clock signal is provided 1<sub>B</sub> The configured external clock signal is provided</p>
<b>NSEL</b>	17	rwh	<p><b>Negation Selection</b></p> <p>0<sub>B</sub> The external clock signal EXTCLK1 is inverted 1<sub>B</sub> The external clock signal EXTCLK1 is not inverted</p>
<b>SEL1</b>	21:18	rwh	<p><b>External Clock Select for EXTCLK1</b></p> <p>This bit field defines the clock source that is selected as the output for pin EXTCLK1.</p> <p>0<sub>H</sub> <math>f_{OUT}</math> is selected for the external clock signal 1<sub>H</sub> <math>f_{PLL0}</math> is selected for the external clock signal 2<sub>H</sub> <math>f_{PLL1}</math> is selected for the external clock signal 3<sub>H</sub> <math>f_{EBU}</math> is selected for the external clock signal 4<sub>H</sub> <math>f_{BACK}</math> is selected for the external clock signal 5<sub>H</sub> <math>f_{MCAN}</math> is selected for the external clock signal 6<sub>H</sub> <math>f_{ADC}</math> is selected for the external clock signal 7<sub>H</sub> <math>f_{QSPI}</math> is selected for the external clock signal 8<sub>H</sub> <math>f_{SRI}</math> is selected for the external clock signal 9<sub>H</sub> <math>f_{SPB}</math> is selected for the external clock signal A<sub>H</sub> <math>f_{I2C}</math> is selected for the external clock signal B<sub>H</sub> <math>f_{MSC}</math> is selected for the external clock signal C<sub>H</sub> <math>f_{ERAY}</math> is selected for the external clock signal D<sub>H</sub> <math>f_{ASCLINF}</math> is selected for the external clock signal E<sub>H</sub> <math>f_{ASCLINS}</math> is selected for the external clock signal F<sub>H</sub> <math>f_{OSCFL}</math> from the flash is selected for the external clock signal</p>
<b>DIV1</b>	31:24	rw	<p><b>External Clock Divider for EXTCLK1</b></p> <p>This value defines the reload value of the divider that generates <math>f_{OUT}</math> out of <math>f_{SPB}</math> (<math>f_{OUT} = f_{SPB}/(DIV1+1)</math>). The divider itself is cleared each time bit EN1 is cleared.</p>
<b>0</b>	1, 15:6, 23:22	r	<p><b>Reserved</b></p> <p>Read as 0; Should be written with 0.</p>

## Clocking System

### Fractional Divider Register

#### FDR

**Fractional Divider Register** **(0038<sub>H</sub>)** **System Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>DISCLK</b>															

rwh		r													rh

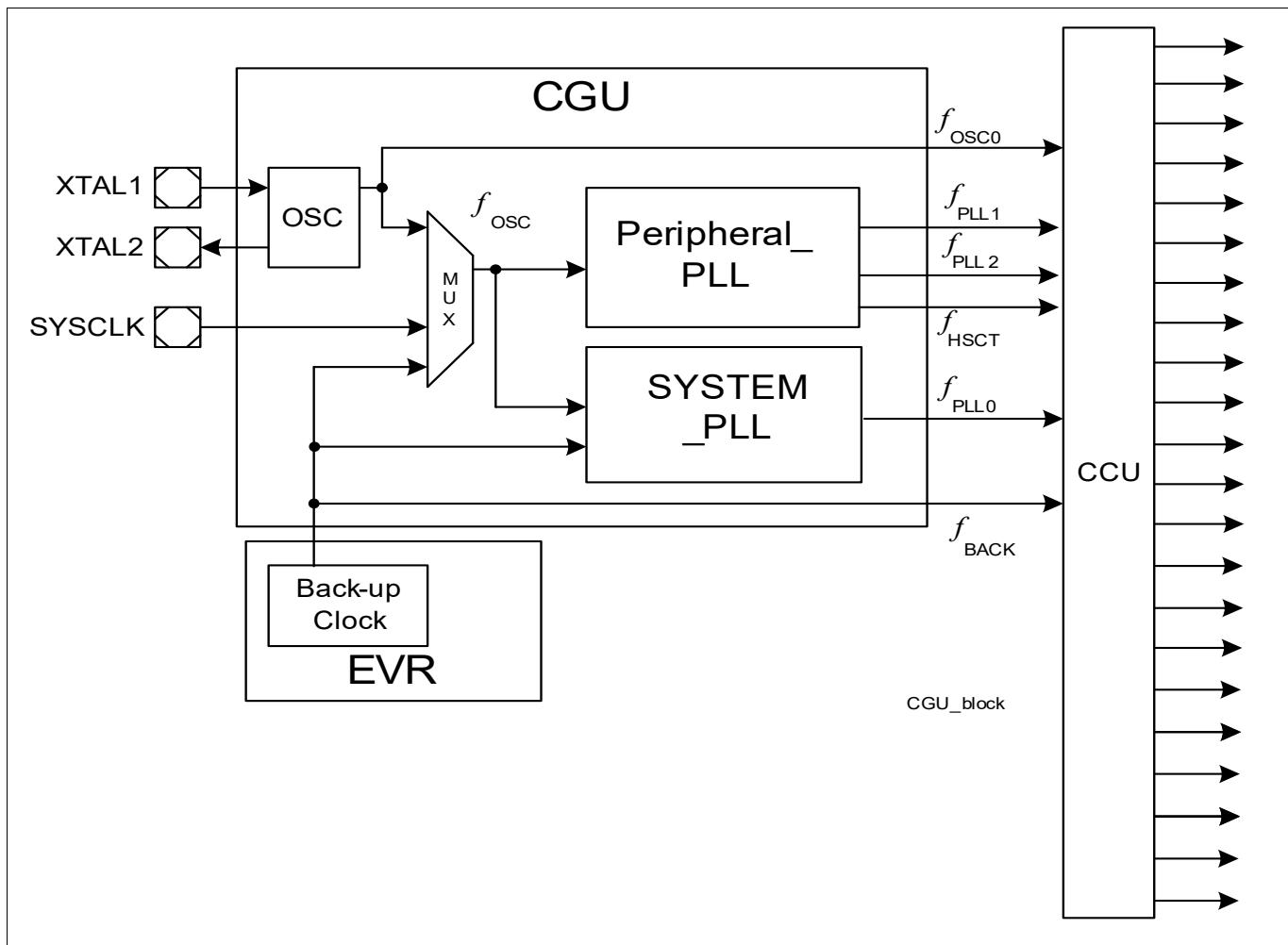
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Field	Bits	Type	Description
<b>STEP</b>	9:0	rw	<b>Step Value</b> In Normal Divider Mode, STEP contains the reload value for RESULT. In Fractional Divider Mode, this bit field determines the 10-bit value that is added to RESULT with each input clock cycle.
<b>DM</b>	15:14	rw	<b>Divider Mode</b> These bit fields determine the functionality of the fractional divider block.  00 <sub>B</sub> Fractional divider is switched off; no output clock is generated. The Reset External Divider signal is 1. RESULT is not updated (default after System Reset). 01 <sub>B</sub> Normal Divider Mode selected. 10 <sub>B</sub> Fractional Divider Mode selected. 11 <sub>B</sub> Fractional divider is switched off; no output clock is generated. RESULT is not updated.
<b>RESULT</b>	25:16	rh	<b>Result Value</b> In Normal Divider Mode, RESULT acts as reload counter (addition +1). In Fractional Divider Mode, this bit field contains the result of the addition RESULT + STEP. If DM is written with 01 <sub>B</sub> or 10 <sub>B</sub> , RESULT is loaded with 3FF <sub>H</sub> .
<b>DISCLK</b>	31	rwh	<b>Disable Clock</b> 0 <sub>B</sub> Clock generation of f <sub>OUT</sub> is enabled according to the setting of bit field DM. 1 <sub>B</sub> Fractional divider is stopped. No change except when writing bit field DM.
<b>0</b>	13:10, 30:26	r	<b>Reserved</b> Read as 0; Should be written with 0.

## Clocking System

### 10.8 Clock Generation Unit

The Clock Generation Unit (CGU) allows a very flexible clock generation for the device. During user program execution, the frequency can be programmed for an optimal ratio between performance and power consumption.



**Figure 89 Clock Generation Unit Block Diagram**

The CGU includes clock source generation and clock up-scaling in addition to clock distribution.

For additional information, see the example sequence in [Chapter 10.10](#).

### 10.9 Safety Measures

#### 10.9.1 Clock Monitoring

For safety, clock alive monitors are available. For the following safety relevant clocks in the system, monitors are present:

- $f_{PLL0}$
- $f_{PLL1}$
- $f_{PLL2}$
- $f_{SPB}$

## Clocking System

- $f_{\text{BACK}}$

Each of these clocks is monitored by its own counter. For the PLL and SPB clocks, the back-up clock is used as a diverse clock source as a monitoring/checking clock. For monitoring the back-up clock,  $f_{\text{PLL0}}$  is used as the diverse clock.

The basic principle of alive monitoring is to detect that the monitored clock is toggling within a certain reference time slot generated by the diverse, observing/monitoring clock. If the monitored clock toggles, it is considered as alive.

The monitored clocks shall be divided down by a certain factor to deliver a lower frequency signal that can be properly sampled in the monitoring clock domain. I.e. division must be tailored in a way that the generated signal has lower frequency than the sampling/monitoring clock, but toggles at least once within the monitor reference time window. The division factor must also consider the allowed clock frequency ranges of both monitored and monitoring clocks and their effective combinations.

The monitoring clocks generate a trigger pulses every 512 clock cycles. With every trigger, the toggle check is performed again and if it fails, an SMU alarm is generated. Hence, the fault reaction time correlates with the duration of the reference timing window.

To ensure safe operation of the EVR33 and EVRC regulators, the backup clock is additionally monitored, as it is the operating clock of these blocks. To adapt to  $f_{\text{PLL0}}$  which is the diverse clock of that monitor, upper and lower monitoring thresholds need to be set accordingly.

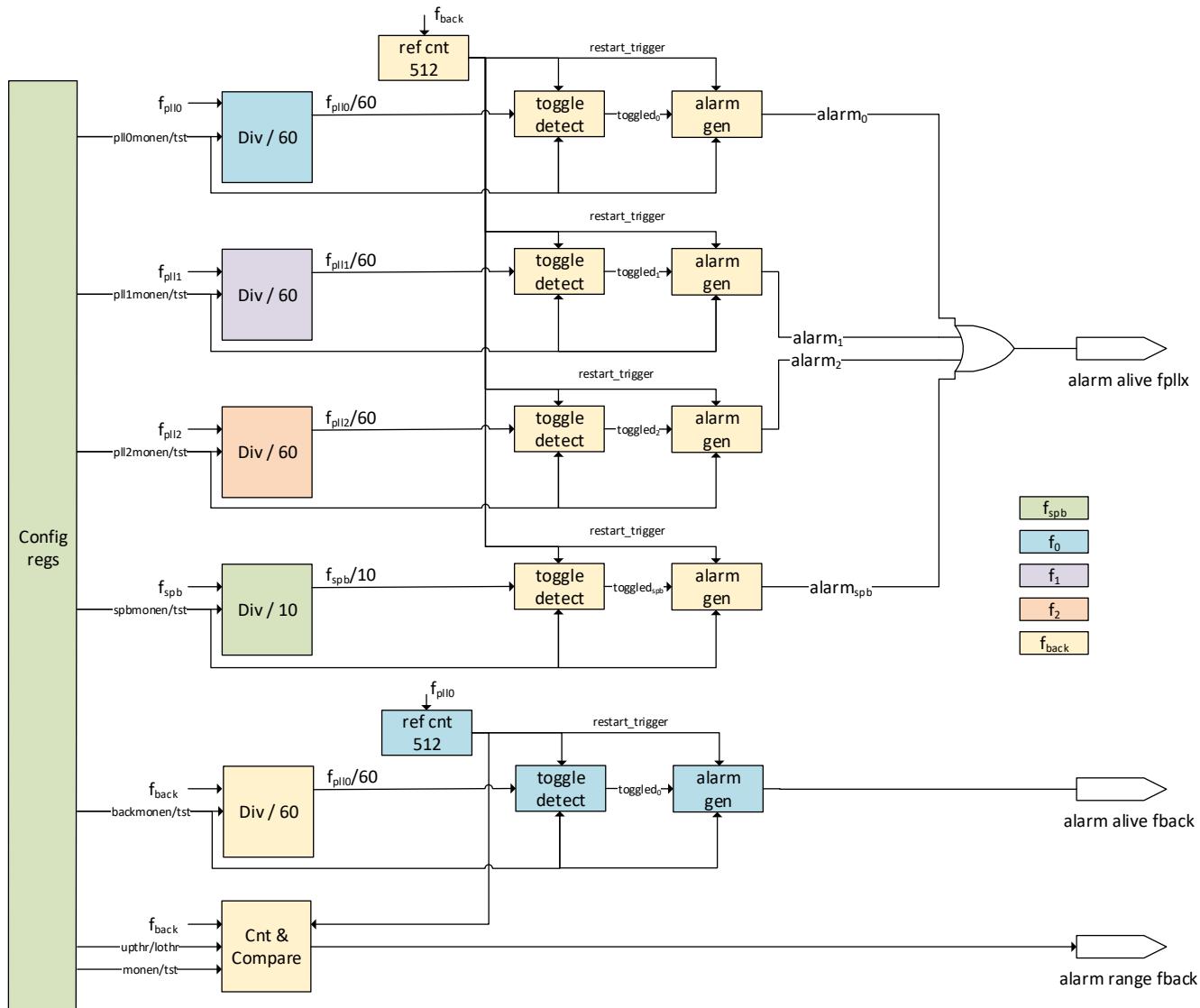
The backup clock monitor simply counts the number of backup clock cycles within a time window of 512  $f_{\text{PLL0}}$  clock cycles. If the number of counted cycles is either above the upper or below the lower threshold, an alarm is generated. This check is continuously repeated.

Clock monitor alarm generation is of level type; i.e., as long as the error condition persists (clock not alive or out of bounce), the alarm is asserted. If the error condition resolves until the next reference cycle restart (trigger), the alarm gets de-asserted again. This assures alarm generation to be long enough for the SMU to process it, i.e. at least one reference counter cycle.

In addition, all monitors can be enabled/disabled and tested via configuration registers. If the monitors get disabled, they will clear all internal status flags and counters to avoid false alerts when the monitor is enabled again.

Because configuration registers, monitored clock dividers, and monitoring logic are asynchronous to each other, synchronization logic has been inserted between the components.

## Clocking System



**Figure 90 Clock Monitor**

The clock alive counters generate two separate alarms; the back-up clock alive monitor generates an alarm signal inside the SPB clock domain, and the  $f_{PLL0}$ ,  $f_{PLL1}$ ,  $f_{PLL2}$  and  $f_{SPB}$  clock alive monitors generate a combined alarm signal inside the Back-up clock domain. The additional backup clock monitor generates an alarm within the SPB clock domain.

## Clocking System

### 10.9.1.1 Clock Monitor Registers

The Clock monitor registers can be accessed by all CPUs in the system. However, it is suggested that only one CPU is used to control the clocks. As CPU0 is the active and available CPU after each reset, this is the best choice.

#### CCU Clock Control Register 3

**CCUCON3**

**CCU Clock Control Register 3**

(0044<sub>H</sub>)

**System Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	<b>UP</b>				<b>0</b>						<b>0</b>				
rh	w			r							rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			<b>BACK</b>	<b>SPBM</b>	<b>PLL2M</b>	<b>PLL1M</b>	<b>PLL0M</b>				<b>BACK</b>	<b>SPBM</b>	<b>PLL2M</b>	<b>PLL1M</b>	<b>PLL0M</b>
			<b>MONT</b>	<b>ONTS</b>	<b>ONTS</b>	<b>ONTS</b>	<b>ONTS</b>				<b>MONE</b>	<b>ONEN</b>	<b>ONEN</b>	<b>ONEN</b>	<b>ONEN</b>
rw			rw	rw	rw	rw	rw				rw	rw	rw	rw	rw

Field	Bits	Type	Description
<b>PLL0MONEN</b>	0	rw	<b>PLL0 Clock Monitor Enable</b> 0 <sub>B</sub> Monitoring is disabled 1 <sub>B</sub> Monitoring is enabled.
<b>PLL1MONEN</b>	1	rw	<b>PLL1 Clock Monitor Enable</b> 0 <sub>B</sub> Monitoring is disabled 1 <sub>B</sub> Monitoring is enabled.
<b>PLL2MONEN</b>	2	rw	<b>PLL2 Clock Monitor Enable</b> 0 <sub>B</sub> Monitoring is disabled 1 <sub>B</sub> Monitoring is enabled.
<b>SPBMONEN</b>	3	rw	<b>SPB Clock Monitor Enable</b> 0 <sub>B</sub> Monitoring is disabled 1 <sub>B</sub> Monitoring is enabled.
<b>BACKMONEN</b>	4	rw	<b>Backup Clock Monitor Enable</b> 0 <sub>B</sub> Monitoring is disabled 1 <sub>B</sub> Monitoring is enabled.
<b>PLL0MONTST</b>	8	rw	<b>PLL0 Clock Monitor Test</b> The test enable bit is not a direct trigger for the alarm, but an inhibit for the clock to be monitored. This is to test the monitoring logic itself as well. 0 <sub>B</sub> normal operation 1 <sub>B</sub> Inhibit $f_{PLL0}$ at monitor input. It may take a full monitor reference count period (512 $f_{BACK}$ cycles) until an alarm is generated. This depends on the selected PLL frequencies.

## Clocking System

Field	Bits	Type	Description
<b>PLL1MONTST</b>	9	rw	<p><b>PLL1 Clock Monitor Test</b></p> <p>The test enable bit is not a direct trigger for the alarm, but an inhibit for the clock to be monitored. This is to test the monitoring logic itself as well.</p> <p>0<sub>B</sub> normal operation 1<sub>B</sub> Inhibit <math>f_{PLL1}</math> at monitor input. It may take a full monitor reference count period (512 <math>f_{BACK}</math> cycles) until an alarm is generated. This depends on the selected PLL frequencies.</p>
<b>PLL2MONTST</b>	10	rw	<p><b>PLL2 Clock Monitor Test</b></p> <p>The test enable bit is not a direct trigger for the alarm, but an inhibit for the clock to be monitored. This is to test the monitoring logic itself as well.</p> <p>0<sub>B</sub> normal operation 1<sub>B</sub> Inhibit <math>f_{PLL2}</math> at monitor input. It may take a full monitor reference count period (512 <math>f_{BACK}</math> cycles) until an alarm is generated. This depends on the selected PLL frequencies.</p>
<b>SPBMONTST</b>	11	rw	<p><b>SPB Clock Monitor Test</b></p> <p>The test enable bit is not a direct trigger for the alarm, but an inhibit for the clock to be monitored. This is to test the monitoring logic itself as well.</p> <p>0<sub>B</sub> normal operation 1<sub>B</sub> Inhibit <math>f_{SPB}</math> at monitor input. It may take a full monitor reference count period (512 <math>f_{BACK}</math> cycles) until an alarm is generated. This depends on the selected PLL frequencies.</p>
<b>BACKMONTST</b>	12	rw	<p><b>Backup Clock Monitor Test</b></p> <p>The test enable bit is not a direct trigger for the alarm, but an inhibit for the clock to be monitored. This is to test the monitoring logic itself as well.</p> <p>0<sub>B</sub> normal operation 1<sub>B</sub> Inhibit <math>f_{BACK}</math> at monitor input. It may take a full monitor reference count period (512 <math>f_{PLL0}</math> cycles) until an alarm is generated. This depends on the selected PLL frequencies.</p>
<b>UP</b>	30	w	<p><b>Update Request</b></p> <p>Setting this bit will request an update for CCUCON3 and CCUCON4. Only one UP bit must be set for either CCUCON3 or CCUCON4. This bit always reads as zero.</p> <p>0<sub>B</sub> No action 1<sub>B</sub> A new complete parameter set is transferred to the CCU defined by register CCUCON3 and CCUCON4.</p>

## Clocking System

Field	Bits	Type	Description
LCK	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.  <i>Note:</i> <i>The lock bit is set when an update of CCUCON3/4 has been requested, and released when the update is complete.</i>  0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and can not be updated
0	7:5, 23:13	rw	<b>Reserved</b> Should be written with 0.
0	29:24	r	<b>Reserved</b> Read as 0; Should be written with 0.

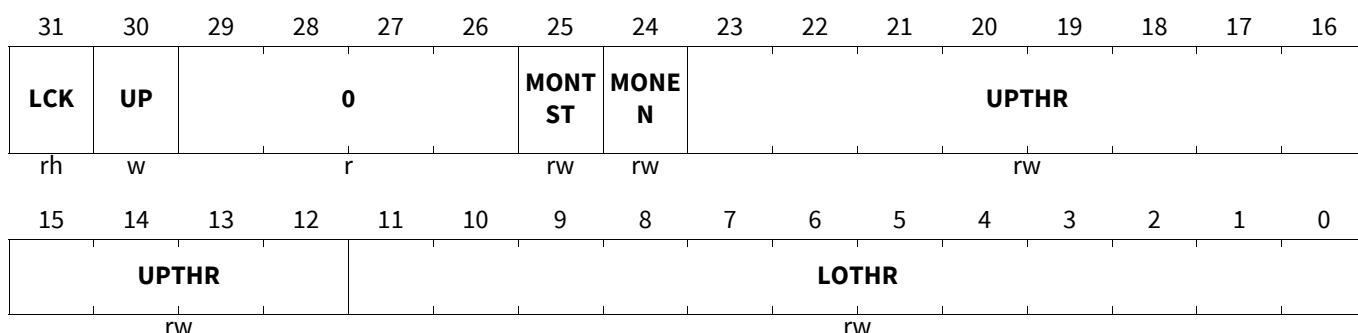
### CCU Clock Control Register 4

**CCUCON4**

**CCU Clock Control Register 4**

**(0048<sub>H</sub>)**

**System Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
LOTHR	11:0	rw	<b>Backup Clock Monitor Lower Threshold</b> lower threshold = $512/f_{\text{PLL0}} * 0.9 * 100 \text{ MHz}$  <i>Note:</i> <i>For proper operation and to avoid false alarms, the monitor needs to be disabled via MONEN=0 before changing/setting the threshold values.</i>
UPTHR	23:12	rw	<b>Backup Clock Monitor Upper Threshold</b> upper threshold = $512/f_{\text{PLL0}} * 1.1 * 100 \text{ MHz}$  <i>Note:</i> <i>For proper operation and to avoid false alarms, the monitor needs to be disabled via MONEN=0 before changing/setting the threshold values.</i>
MONEN	24	rw	<b>Backup Clock Monitor Enable</b> 0 <sub>B</sub> Monitoring is disabled 1 <sub>B</sub> Monitoring is enabled

## Clocking System

Field	Bits	Type	Description
MONTST	25	rw	<b>Backup Clock Monitor Test</b> Set this bit to 1 to test alarm generation. The test enable bit is a direct trigger for the alarm. 0 <sub>B</sub> Normal Operation 1 <sub>B</sub> Test Alarm will be generated
UP	30	w	<b>Update Request</b> Setting this bit will request an update for CCUCON3 and CCUCON4. Only one UP bit must be set for either CCUCON3 or CCUCON4. This bit always reads as zero. 0 <sub>B</sub> No action 1 <sub>B</sub> A new complete parameter set is transferred to the CCU defined by register CCUCON3 and CCUCON4.
LCK	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.  <i>Note:</i> <i>The lock bit is set when an update of CCUCON3/4 has been requested, and released when the update is complete.</i> 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and can not be updated
0	29:26	r	<b>Reserved</b> Read as 0; Should be written with 0.

## 10.10 Use Cases

### Clock Ramp-up Example

The following sequence gives an example for the clock ramp-up. The goal for normal target setting is that the system clock is based on the PLL with an external crystal.

- After power-on and system reset:
  - the system operates on the back-up clock
  - the oscillator (OSC) needs to be enabled via software or can be enabled by firmware via BMI settings
- Fast clocks (SRI-Bus and CPUs) run at the back-up clock frequency (trimmed to approximately 100 MHz)
- Peripherals and SPB-Bus run at 1/3 of the back-up clock frequency (trimmed to approximately 33 MHz)
- In the following steps, two settings AAA and BBB are described with the notation: AAA[BBB]  
 Setting AAA is defined for a 20MHz crystal / clock input and is configured for a 300MHz system.  
 Setting BBB is defined for a 25MHz crystal / clock input and is configured for a 300MHz system.
- Step 1: If  $f_{\text{OSC}0}$  is to be used but is not running, enable the oscillator and wait until it is providing a stable clock.
- Step 2: Initialize the PLLs to target  $f_{\text{DCO}}$  and  $f_{\text{PLLX}}$  frequency
  - Select PLL input clock via SYSPLLCON0.INSEL = 01<sub>B</sub>
  - Select P, K2, and N divider for final target DCO and PLL frequency  
 $\text{SYSPLLCON0} = 40013A00_H[40012E00_H]$   
 $\text{SYSPLLCON1} = 00000005_H[00000005_H]$   
 $\text{SYSPLL } f_{\text{DCO}} = 600\text{MHz}[600\text{MHz}]; f_{\text{PLL0}} = 100\text{MHz}[100\text{MHz}]$   
 $\text{PERPLLCON0} = 00013E00_H[00013E01_H]$

## Clocking System

`PERPLLCON1 = 00000101H[00000104H]`

$f_{DCO} = 640\text{MHz}[800\text{MHz}]$ ;  $f_{PLL1} = 320\text{MHz}[160\text{MHz}]$ ;  $f_{PLL2} = 200\text{MHz}[200\text{MHz}]$

- Step 3: Wait for PLL lock to be set
- Step 4: Configure CCUCON0, and CCUCON1 to first target setting  
 $\text{CCUCON0} = 07230113_{H}[07230113_{H}]$   
 $\text{CCUCON1} = 21210312_{H}[21210392_{H}]$   
 $\text{CCUCON2} = 07001201_{H}[07001201_{H}]$   
 $\text{CCUCON5} = 40000132_{H}[40000030_{H}]$   
 $f_{STM} = 33.3\text{MHz}[33.3\text{MHz}]$ ;  $f_{GTM} = 66.6\text{MHz}[66.6\text{MHz}]$ ;  $f_{SRI} = 100\text{MHz}[150\text{MHz}]$ ;  $f_{SPB} = 33.3\text{MHz}[33.3\text{MHz}]$ ;  
 $f_{BBB} = 50\text{MHz}[50\text{MHz}]$ ;  $f_{FSI} = 33.3\text{MHz}[33.3\text{MHz}]$ ;  $f_{FSI2} = 100\text{MHz}[100\text{MHz}]$ ;  
 $f_{GETH} = 50\text{MHz}[50\text{MHz}]$ ;  $f_{MCANH} = 33.3\text{MHz}[33.3\text{MHz}]$ ;  $f_{ADAS} = 100\text{MHz}[100\text{MHz}]$ ;  
 $f_{MCAN} = 50\text{MHz}[50\text{MHz}]$ ;  $f_{I2C} = 33.3\text{MHz}[33.3\text{MHz}]$ ;  $f_{MSC} = 100\text{MHz}[100\text{MHz}]$ ;  $f_{QSPI} = 100\text{MHz}[100\text{MHz}]$ ;  
 $f_{ASCLINF} = 100\text{MHz}[100\text{MHz}]$ ;  $f_{ASCLINS} = 50\text{MHz}[50\text{MHz}]$ ;  $f_{EBU} = 100\text{MHz}[100\text{MHz}]$ ;  $f_{ERAY} = 50\text{MHz}[50\text{MHz}]$ ;  
 $f_{HSPDM\_320} = 100\text{MHz}[100\text{MHz}]$ ;  $f_{HSPDM\_160} = 100\text{MHz}[100\text{MHz}]$ ;  
 $f_{ADC} = 100\text{MHz}[100\text{MHz}]$ ;  $f_{HSCT} = 320\text{MHz}[400\text{MHz}]$ ;  $f_{RAM} = 66.6\text{MHz}[66.6\text{MHz}]$ ;  
 $f_{REFCLK1} = 4.16\text{MHz}[4.16\text{MHz}]$ ;  $f_{REFCLK2} = 4.16\text{MHz}[4.16\text{MHz}]$ ;
- Step 5: Switch CCU input clock  $f_{SOURCE0}$  to PLL via CCUCON0.CLKSEL  
 $\text{CCUCON0} = 57230113_{H}[57230113_{H}]$   
 $f_{STM} = 33.3\text{MHz}[33.3\text{MHz}]$ ;  $f_{GTM} = 66.6\text{MHz}[66.6\text{MHz}]$ ;  $f_{SRI} = 100\text{MHz}[100\text{MHz}]$ ;  $f_{SPB} = 33.3\text{MHz}[33.3\text{MHz}]$ ;  
 $f_{BBB} = 50\text{MHz}[50\text{MHz}]$ ;  $f_{FSI} = 33.3\text{MHz}[33.3\text{MHz}]$ ;  $f_{FSI2} = 100\text{MHz}[100\text{MHz}]$ ;  
 $f_{GETH} = 50\text{MHz}[50\text{MHz}]$ ;  $f_{MCANH} = 33.3\text{MHz}[33.3\text{MHz}]$ ;  $f_{ADAS} = 100\text{MHz}[100\text{MHz}]$ ;  
 $f_{MCAN} = 80\text{MHz}[80\text{MHz}]$ ;  $f_{I2C} = 66.6\text{MHz}[66.6\text{MHz}]$ ;  $f_{MSC} = 200\text{MHz}[200\text{MHz}]$ ;  $f_{QSPI} = 200\text{MHz}[200\text{MHz}]$ ;  
 $f_{ASCLINF} = 200\text{MHz}[200\text{MHz}]$ ;  $f_{ASCLINS} = 80\text{MHz}[80\text{MHz}]$ ;  $f_{EBU} = 160\text{MHz}[160\text{MHz}]$ ;  $f_{ERAY} = 80\text{MHz}[80\text{MHz}]$ ;  
 $f_{HSPDM\_320} = 320\text{MHz}[160\text{MHz}]$ ;  $f_{HSPDM\_160} = 160\text{MHz}[160\text{MHz}]$ ;  
 $f_{ADC} = 160\text{MHz}[160\text{MHz}]$ ;  $f_{HSCT} = 320\text{MHz}[400\text{MHz}]$ ;  $f_{RAM} = 66.6\text{MHz}[66.6\text{MHz}]$ ;  
 $f_{REFCLK1} = 4.16\text{MHz}[4.16\text{MHz}]$ ;  $f_{REFCLK2} = 6.67\text{MHz}[6.67\text{MHz}]$ ;
- Step 6: After setting CCU  $f_{SOURCE}$  to  $f_{PLL}$ , the frequency has to be increased step by step to the final target frequency
  - a)  $\text{SYSPLLCON1} = 00000003_{H}[00000003_{H}]$   
 $f_{DCO} = 600\text{MHz}[600\text{MHz}]$ ;  $f_{PLL0} = 150\text{MHz}[150\text{MHz}]$   
 $f_{STM} = 50\text{MHz}[50\text{MHz}]$ ;  $f_{GTM} = 100\text{MHz}[100\text{MHz}]$ ;  $f_{SRI} = 150\text{MHz}[150\text{MHz}]$ ;  $f_{SPB} = 50\text{MHz}[50\text{MHz}]$ ;  
 $f_{BBB} = 75\text{MHz}[75\text{MHz}]$ ;  $f_{FSI} = 50\text{MHz}[50\text{MHz}]$ ;  $f_{FSI2} = 150\text{MHz}[150\text{MHz}]$ ;  
 $f_{GETH} = 75\text{MHz}[75\text{MHz}]$ ;  $f_{MCANH} = 50\text{MHz}[50\text{MHz}]$ ;  $f_{ADAS} = 150\text{MHz}[150\text{MHz}]$ ;  
 $f_{MCAN} = 80\text{MHz}[80\text{MHz}]$ ;  $f_{I2C} = 66.6\text{MHz}[66.6\text{MHz}]$ ;  $f_{MSC} = 200\text{MHz}[200\text{MHz}]$ ;  $f_{QSPI} = 200\text{MHz}[200\text{MHz}]$ ;  
 $f_{ASCLINF} = 200\text{MHz}[200\text{MHz}]$ ;  $f_{ASCLINS} = 80\text{MHz}[80\text{MHz}]$ ;  $f_{EBU} = 160\text{MHz}[160\text{MHz}]$ ;  $f_{ERAY} = 80\text{MHz}[80\text{MHz}]$ ;  
 $f_{HSPDM\_320} = 320\text{MHz}[160\text{MHz}]$ ;  $f_{HSPDM\_160} = 160\text{MHz}[160\text{MHz}]$ ;  
 $f_{ADC} = 160\text{MHz}[160\text{MHz}]$ ;  $f_{HSCT} = 320\text{MHz}[400\text{MHz}]$ ;  $f_{RAM} = 100\text{MHz}[100\text{MHz}]$ ;  
 $f_{REFCLK1} = 6.25\text{MHz}[6.25\text{MHz}]$ ;  $f_{REFCLK2} = 6.67\text{MHz}[6.67\text{MHz}]$ ;
  - b)  $\text{SYSPLLCON1} = 00000002_{H}[00000002_{H}]$   
 $f_{DCO} = 600\text{MHz}[600\text{MHz}]$ ;  $f_{PLL0} = 200\text{MHz}[200\text{MHz}]$   
 $f_{STM} = 66.6\text{MHz}[66.6\text{MHz}]$ ;  $f_{GTM} = 133.3\text{MHz}[133.3\text{MHz}]$ ;  $f_{SRI} = 200\text{MHz}[200\text{MHz}]$ ;  $f_{SPB} = 66.6\text{MHz}[66.6\text{MHz}]$ ;  
 $f_{BBB} = 100\text{MHz}[100\text{MHz}]$ ;  $f_{FSI} = 66.6\text{MHz}[66.6\text{MHz}]$ ;  $f_{FSI2} = 200\text{MHz}[200\text{MHz}]$ ;  
 $f_{GETH} = 100\text{MHz}[100\text{MHz}]$ ;  $f_{MCANH} = 66.6\text{MHz}[66.6\text{MHz}]$ ;  $f_{ADAS} = 200\text{MHz}[200\text{MHz}]$ ;  
 $f_{MCAN} = 80\text{MHz}[80\text{MHz}]$ ;  $f_{I2C} = 66.6\text{MHz}[66.6\text{MHz}]$ ;  $f_{MSC} = 200\text{MHz}[200\text{MHz}]$ ;  $f_{QSPI} = 200\text{MHz}[200\text{MHz}]$ ;  
 $f_{ASCLINF} = 200\text{MHz}[200\text{MHz}]$ ;  $f_{ASCLINS} = 80\text{MHz}[80\text{MHz}]$ ;  $f_{EBU} = 160\text{MHz}[160\text{MHz}]$ ;  $f_{ERAY} = 80\text{MHz}[80\text{MHz}]$ ;  
 $f_{HSPDM\_320} = 320\text{MHz}[160\text{MHz}]$ ;  $f_{HSPDM\_160} = 160\text{MHz}[160\text{MHz}]$ ;  
 $f_{ADC} = 160\text{MHz}[160\text{MHz}]$ ;  $f_{HSCT} = 320\text{MHz}[400\text{MHz}]$ ;  $f_{RAM} = 133\text{MHz}[133\text{MHz}]$ ;  
 $f_{REFCLK1} = 8.3\text{MHz}[8.3\text{MHz}]$ ;  $f_{REFCLK2} = 6.67\text{MHz}[6.67\text{MHz}]$ ;
  - c)  $\text{SYSPLLCON1} = 00000001_{H}[00000001_{H}]$

## Clocking System

$f_{\text{SYSPLL } DCO} = 600\text{MHz}[600\text{MHz}]$ ;  $f_{\text{PLL0}} = 300\text{MHz}[300\text{MHz}]$   
 $f_{\text{STM}} = 100\text{MHz}[100\text{MHz}]$ ;  $f_{\text{GTM}} = 200\text{MHz}[200\text{MHz}]$ ;  $f_{\text{SRI}} = 300\text{MHz}[300\text{MHz}]$ ;  $f_{\text{SPB}} = 100\text{MHz}[100\text{MHz}]$ ;  
 $f_{\text{BBB}} = 150\text{MHz}[150\text{MHz}]$ ;  $f_{\text{FSI}} = 100\text{MHz}[100\text{MHz}]$ ;  $f_{\text{FSI2}} = 300\text{MHz}[300\text{MHz}]$ ;  
 $f_{\text{GETH}} = 150\text{MHz}[150\text{MHz}]$ ;  $f_{\text{MCANH}} = 100\text{MHz}[100\text{MHz}]$ ;  $f_{\text{ADAS}} = 300\text{MHz}[300\text{MHz}]$ ;  
 $f_{\text{MCAN}} = 80\text{MHz}[80\text{MHz}]$ ;  $f_{\text{I2C}} = 66.6\text{MHz}[66.6\text{MHz}]$ ;  $f_{\text{MSC}} = 200\text{MHz}[200\text{MHz}]$ ;  $f_{\text{QSPI}} = 200\text{MHz}[200\text{MHz}]$ ;  
 $f_{\text{ASCLINF}} = 200\text{MHz}[200\text{MHz}]$ ;  $f_{\text{ASCLINS}} = 80\text{MHz}[80\text{MHz}]$ ;  $f_{\text{EBU}} = 160\text{MHz}[160\text{MHz}]$ ;  $f_{\text{ERAY}} = 80\text{MHz}[80\text{MHz}]$ ;  
 $f_{\text{HSPDM}_320} = 320\text{MHz}[160\text{MHz}]$ ;  $f_{\text{HSPDM}_160} = 160\text{MHz}[160\text{MHz}]$ ;  
 $f_{\text{ADC}} = 160\text{MHz}[160\text{MHz}]$ ;  $f_{\text{HSCT}} = 320\text{MHz}[400\text{MHz}]$ ;  $f_{\text{RAM}} = 200\text{MHz}[200\text{MHz}]$ ;  
 $f_{\text{REFCLK1}} = 12.5\text{MHz}[12.5\text{MHz}]$ ;  $f_{\text{REFCLK2}} = 6.67\text{MHz}[6.67\text{MHz}]$ ;

Hint for step 4: CCUCON1.PLL1DIVDIS should be set to 0<sub>B</sub> if PERPLLCON1.K2DIV is configured to 0<sub>H</sub> or 1<sub>H</sub>. CCUCON1.PLL1DIVDIS should be set to 1<sub>B</sub> if PERPLLCON1.K2DIV is configured to 2<sub>H</sub> or a bigger value.

Important hint for steps 5 and 6:

- After every frequency programming step, a wait time is recommended until the supply ripple caused by the supply current transient has settled.

Note: *The wait time between frequency steps depends on the supply and block concept.*

### Clock Changing Example

If the setup of the clock system has to be changed, there are two different cases to distinguish between:

- Changing one or more frequencies, only requiring CCUCONx register changes
- Changing frequencies requiring a change in the SYSPLL and / or PERPLL

If only one or several CCUCON registers need to be updated for the new intended configuration, this can directly be done without further preparation.

If one or both PLLs have to be re-configured, some preparation should be done up front to the sequence described above for clock ramp-up. First, the clock system should be configured to operate again on the back-up clock as it does after any system reset. Changing the configuration setting of a PLL while the clock system operates on this clock source is not recommended. Before executing step 2 of the sequence, the PLL needs to be prepared.

If a change for P-divider setting is required, first a value different than the actual configuration should be written to clean the pipeline for the new value. Please note that this is only required for the P-divider, and not for the other PLL dividers or configuration bits.

If for any reason one PLL was set to Power Saving Mode, and thereafter configured to Normal Mode again, the same preparation as already described for the P-divider has to be done for the K- and N-dividers.

## Clocking System

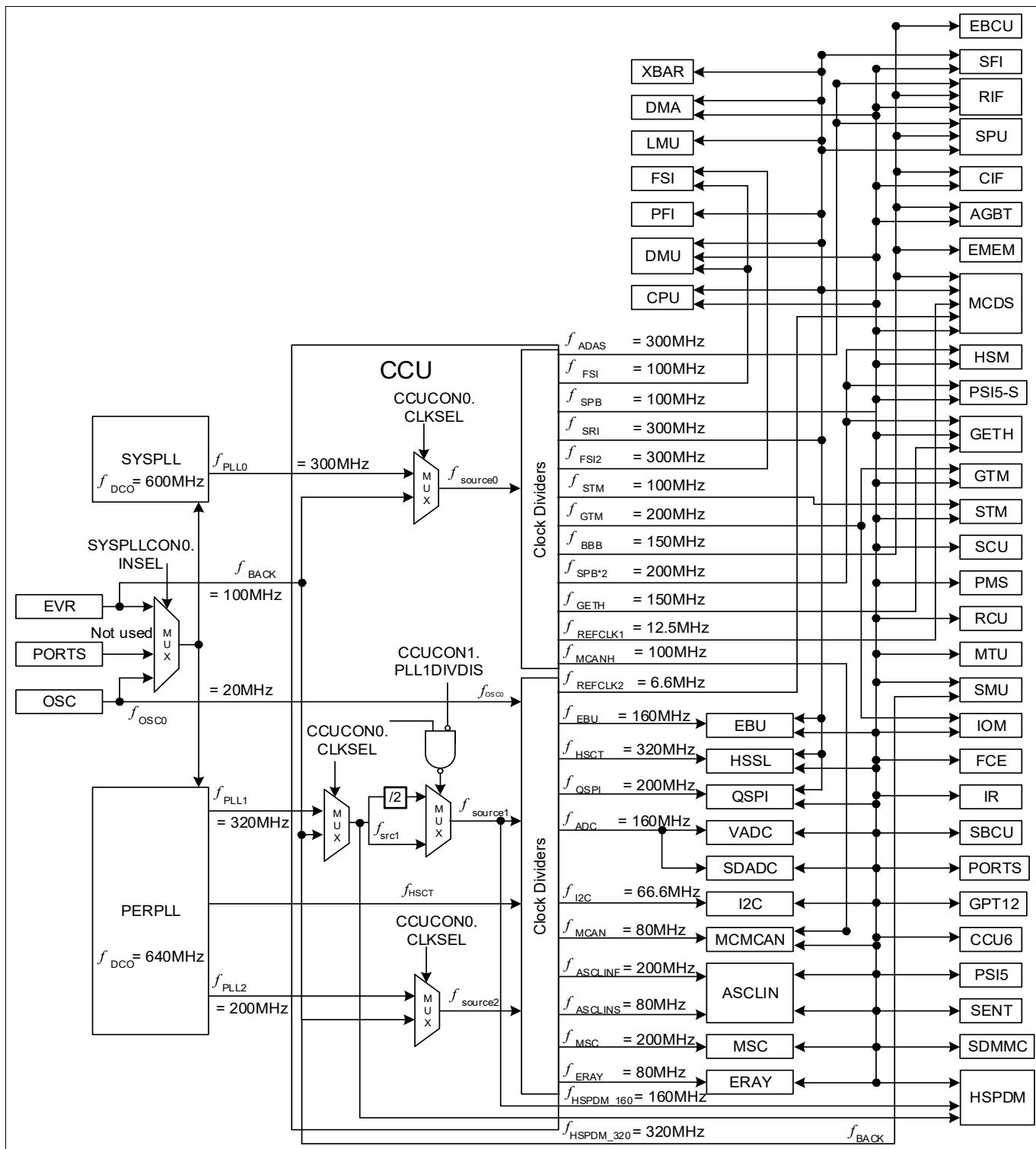


Figure 91 Clocking System example with external 20MHz crystal / clock input

## Clocking System

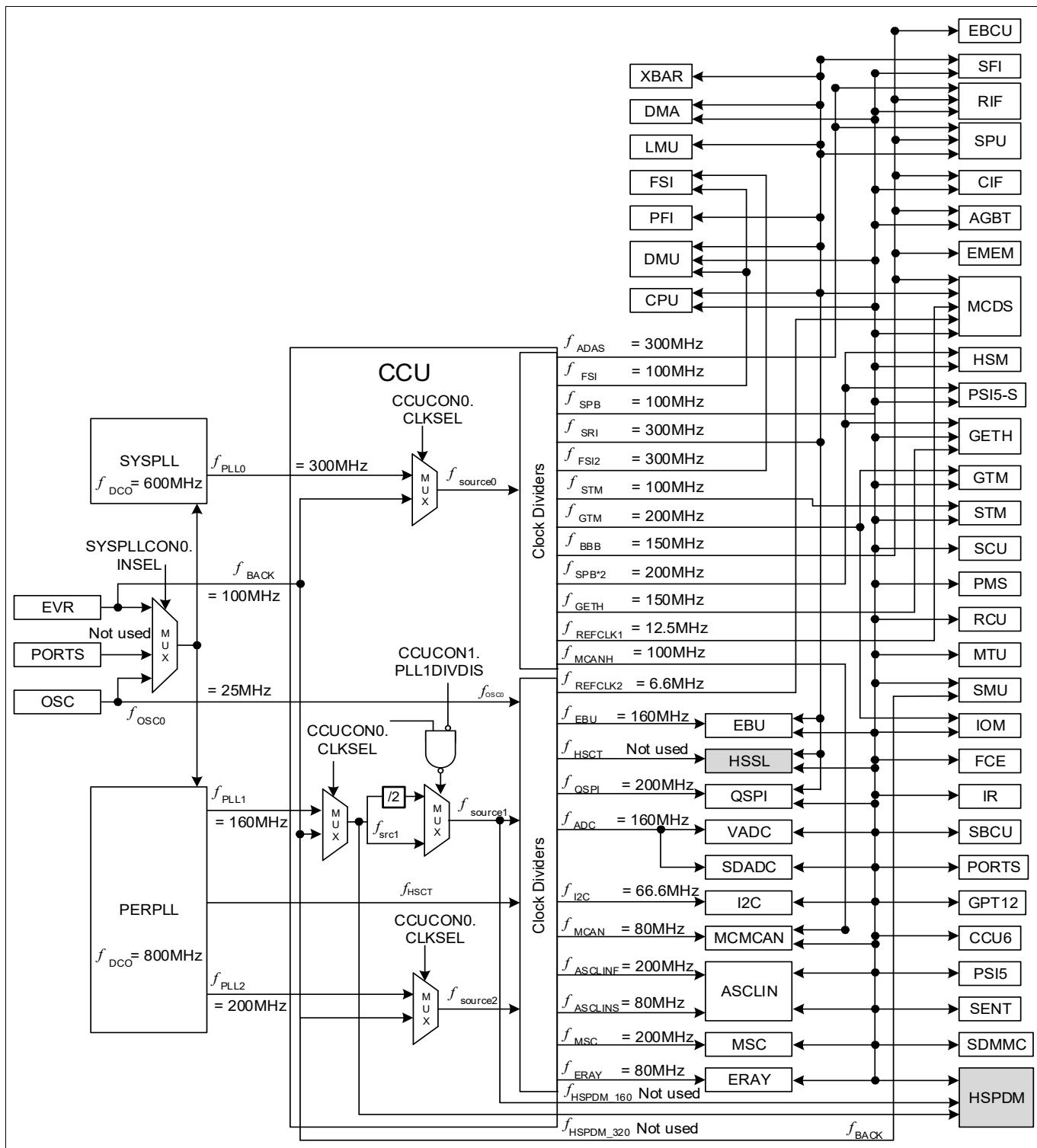


Figure 92 Clocking System example with external 25MHz crystal / clock input

**Clocking System****10.11 Revision History****Table 284 Revision History**

Reference	Change to Previous Version	Comment
<b>V2.0.20</b>		
<a href="#">Page 32</a>	Wording changed for register bitfield description DIVDIS in <b>CCUCON1</b> .	
<a href="#">Page 28,39 ,53,55</a>	Added notes in <b>CCUCON0,CCUCON5,CCUCON3,CCUCON4</b> to describe LOCK mechanism in more detail.	
<b>V2.0.21</b>		
<a href="#">Page 6</a>	Added note about LVDS distributor adjustment in <b>OSCCON.GAINSEL</b> for Vext=3.3V.	
<a href="#">Page 6</a>	Added information how to calculate OSCWD threshold frequencies in <b>OSCCON.PLLLV /PLLHV</b> .	
<a href="#">Page 6</a>	Replaced the word GAIN with GAINSEL in <b>OSCCON.APREN</b> . Added the hint “default” in <b>OSCCON.HYSCTL</b> “00”.	
<a href="#">Page 6</a>	Added the hint “recommended” in <b>OSCCON.HYSEN</b> “1”	
<a href="#">Page 47</a>	Replaced “reserved” fields in <b>EXTCON.SEL0/1</b> with actual clock selection settings.	
<b>V2.0.22</b>		
<a href="#">Page 39</a>	Re-added ADAS DIV settings > 1 / reverted reserved fields <b>CCUCON5</b> .	
<b>V2.0.23</b>		
<a href="#">Page 28</a>	Set <b>CCUCON0.FSI2DIV</b> > 1 to reserved.	
<b>V2.0.24</b>		
<a href="#">Page 27</a>	Removed entries in <b>CCU allowed Clock Ratios</b> n> 1 for SRI/FSI2.	
<a href="#">Page 61</a>	Revision history layout change from multiple tables to one table, hide revisions in revision history lower than V2.0.20	
<a href="#">Page 3</a>	Added SubChapter for Safety FlipFlop listing.	
<a href="#">Page 27</a>	Changed recommended default value from 1 to 2 for SRI/GETH in <b>CCU allowed Clock Ratios</b> .	
<a href="#">Page 6</a>	Removed “external input clock mode” from in <b>OSCCON.MODE</b> field.	
<a href="#">Page 6</a>	Added extra explanation how to read the formulas in <b>OSCCON.PLLLV/PLLHV</b> , added NDIV as variable.	
<b>V2.0.25</b>		
<a href="#">Page 6</a>	removed NDIV from formulas in <b>OSCCON.PLLLV/PLLHV</b>	
<b>V2.0.26</b>		
<a href="#">Page 6</a>	Added comments “reserved/default” to <b>OSCCON.SHBY</b> bit description.	

## Clocking System

**Table 284 Revision History (cont'd)**

Reference	Change to Previous Version	Comment
<a href="#">Page 56</a>	Corrected CCUCON0/5 programming steps in Clock Ramp-Up Example in <a href="#">Chapter 10.10</a> .	
<a href="#">Page 28Page 39</a>	Corrected <a href="#">CCUCON0</a> system reset value. Removed separate Reset Table for <a href="#">CCUCON5</a> .	
<b>V2.0.27</b>		
<a href="#">Page 56</a>	Removed listing of HSM frequency in Clock programming example as it is an IFX internal design name.	
<a href="#">Page 52</a>	Corrected divider factor in SPB alive monitor from 60 to 10 in <a href="#">Figure 90</a>	
<b>V2.0.28</b>		
<a href="#">Page 24</a>	Removed phrase “and PSI5S” from QSPI frequency description in <a href="#">Basic Clock System Mechanisms</a> .	
<b>V2.0.29</b>		
<a href="#">Page 23</a>	Removed additional “?” after first paragraph in <a href="#">Chapter 10.5</a> which was inserted by accident.	
<a href="#">Page 28Page 39</a>	Replaced enumeration Value ”110” in register <a href="#">CCUCON0.LPDIV</a> and “D” in register <a href="#">CCUCON5.GETHDIV</a> with “...” .	
<a href="#">Page 52</a>	Corrected <a href="#">Figure 90</a> again as the old one was placed in version 2.0.28 by accident.	
<b>V2.0.30</b>		
<a href="#">Page 9</a>	Added note for watchdog usage and related alarm settings depending on selected input clock for oscillator in <a href="#">Chapter 10.3.1.5</a>	

## Power Management System (PMS)

# 11 Power Management System (PMS)

This chapter describes Power Supply Generation and Power Management in TC3xx in following sections:

- Power Supply Infrastructure and Supply Start-up (see [Section 11.2.1](#))
  - Supply Mode Selection (see [Section 11.2.1.1](#))
  - Supply Ramp-up and Ramp-down Behavior (see [Section 11.2.1.2](#))
  - Independent Supply domain for Regulators and Monitors (see [Section 11.2.1.3.1](#))
  - Reference Voltage Generation (see [Section 11.2.1.3.2](#))
  - 100 MHz Back-up Clock (see [Section 11.2.1.3.3](#))
  - Die Temperature Sensor (DTS) (see [Section 11.2.1.4](#))
- Power Supply Generation and Monitoring (see [Section 11.2.2](#))
  - VDDP3 Supply Generation
    - Linear Regulator Mode (EVR33) (see [Section 11.2.2.1](#))
    - External Supply Modes (see [Section 11.2.2.4](#))
  - VDD Supply Generation
    - Step-down Regulator (EVRC) (see [Section 11.2.2.2](#))
    - External Supply Modes (see [Section 11.2.2.4](#))
  - Supply Voltage Monitoring (see [Section 11.2.2.5](#))
    - Primary under-voltage monitors and Cold PORST (see [Section 11.2.2.5.1](#))
    - Secondary over- and under-voltage monitors and alarm generation (see [Section 11.2.2.5.2](#))
    - Built In Self Tests (PBIST and MONBIST) (see [Section 11.2.2.5.3](#) and [Section 11.2.2.5.4](#))
  - Interrupts (see [Section 11.2.2.6](#))
  - OCDS Interface (see [Section 11.2.2.7](#))
- Power Management (see [Section 11.2.3](#))
  - Idle Mode (see [Section 11.2.3.2](#))
  - Sleep Mode (see [Section 11.2.3.3](#))
  - Standby Mode (see [Section 11.2.3.4](#))
  - Wake-up Timer (WUT) (see [Section 11.2.3.4.7](#))
  - Standby ControlleR (SCR) Interface (see [Section 11.2.3.4.6](#))
  - Load Jump Sequencing and Voltage Droop (see [Section 11.2.3.5](#))
- Power Management System Register Tables
  - PMS Power Management Register Table (see [Page 79](#))
  - SCU Power Management Register Table (see [Page 180](#))

## Power Management System (PMS)

### 11.1 Overview

On-chip linear and switch mode voltage regulators are implemented in TC3xx thereby enabling a single source power supply concept. The external nominal system supply from external regulator may be either 5 V or 3.3 V. The Embedded Voltage Regulators (EVR33 & EVRC) in turn generate the VDDP3 and VDD supply voltages required internally for the core, flash and port domains. EVRC regulator is implemented as a SMPS regulator and generates core supply either from 5 V or 3.3 V external supply. EVR33 regulator is implemented always as a LDO regulator and is required only in case of 5 V external supply.

Depending on the chosen EVR mode, the actual power consumption, EMI requirements and thermal constraints of the system; additional external components like MOSFETs, inductors and capacitors may be required. It is also possible to supply all voltages (VEXT, VDDP3 and VDD) externally ensuring compliance to the legacy supply concept.

All supply and generated voltages are monitored for brownout conditions by primary monitors setting the device into cold power-on reset state in case of violation. All supply and generated voltages are monitored again redundantly by secondary monitors against programmable over-voltage and under-voltage levels. If these levels are violated, either an interrupt or an alarm to the SMU may be generated.

All internal supplies except analog supplies (VAREFx & VDDM) may be supplied by the EVR33 & EVRC. The analog supply domain is separated from the main EVR supply domain and can be supplied by separate external regulators or trackers. It is possible to have a mixed supply scheme with a 5 V ADC domain (VAREFx = VDDM = 5 V) and the remaining system running on 3.3 V supply (VEXT = VDDP3 = 3.3 V).

### 11.2 Functional Description

#### 11.2.1 Power Supply Infrastructure and Supply Start-up

##### 11.2.1.1 Supply Mode Selection

The choice of the supply scheme at startup is based on the latched status of HWCFG[2:1] pins before cold PORST release and is indicated by [PMSWSTAT](#).HWCFGEVR status flags. Following supply modes are supported and are further enumerated in [Table 285](#).

- Single source 5 V supply level ( $VEXT = 5\text{ V}$ ) is supported in following topologies.
  - EVRC in SMPS mode with external switches and EVR33 in LDO mode with internal pass devices.
- Single source 3.3 V supply level ( $VEXT = VDDP3 = 3.3\text{ V}$ ) is supported in following topologies.
  - EVRC in SMPS mode with external switches and EVR33 is inactive.
- Supplies are provided externally and the respective EVRs are in disabled state.
  - 5 V ( $VEXT$ ) and 1.25 V ( $VDD$ ) supplied externally. EVR33 in LDO mode with internal pass devices.
  - 5 V ( $VEXT$ ) and 3.3 V ( $VDDP3$ ) supplied externally. EVRC in SMPS mode with external switches.
  - 5 V ( $VEXT$ ), 3.3 V ( $VDDP3$ ) and 1.25 V ( $VDD$ ) are all supplied externally.

EVRC is enabled or disabled at startup via the HWCFG[2] configuration pin. In case EVRC is selected, VGATE1P and VGATE1N pins shall be connected to the gate of an external P-channel MOSFET and N-channel MOSFET respectively as shown in [Figure 103](#).

## Power Management System (PMS)

EVR33 is enabled or disabled at startup via the HWCFG[1] configuration pin. In case of single source 3.3 V supply, EVR33 is disabled and VDDx3 & VEXT pins are supplied externally by 3.3 V. EVR33 LDO uses internal pass devices distributed on the chip.

The allowed ranges of supplies among different supply rails during different power modes are documented in [Table 286](#) and [Table 287](#). The allowed combinations of nominal external supply voltages among different supply rails are documented in [Table 288](#). All externally provided supplies must be available and be stable before warm PORST reset release by external regulator(s).

HWCFG [2:1] are latched during supply ramp-up and the respective regulators are consequently started. The latched values are stored in PMSWSTAT.HWCFCGEVR register bits. The latched values are retained through a cold PORST and are only reset if EVR LVD (Low Voltage Detector) reset is asserted. HWCFG signals are filtered through a spike / glitch filter and are monitored for a constant level over a 28us - 115us nominal debouncing period before the value is considered as valid so as to ensure reliable operation in noisy environment. The current state of EVRs are reflected in EVRSTAT.EVRx3 flags. For small package variants, some of the HWCFG configuration pins may be absent and both EVRs are activated by default at startup.

HWCFG[6] pin is latched during early VEXT supply ramp-up ( $VEXT < VDDPPA$ ) to decide and set the default reset state of port pins as early as possible. During the intial ramp-up phase of VEXT and VEVRSB supply voltage from 0V up to VDDPPA limit, the voltage levels of pins are undefined till the transistor threshold voltages are reached. After VDDPPA limit, the pins behave as inputs with pull-up if HWCFG[6] = 1 or are in tristate if HWCFG[6] = 0. During the later stage of ramp-up, the latched HWCFG[6] value is stored in PMSWSTAT.TRIST register bit.

HWCFG [1,2,3,6] pins have weak internal pull-up active at start-up irrespective of HWCFG [6] pin level to ensure that the device boots with a defined configuration if HWCFG [1,2,3,6] pins are left unconnected. HWCFG [1,2,3,6] pins are only latched by the PMS on every initial supply ramp-up and are not re-latched during warm reset events (warm PORST, system or application resets) or on exit from Standby mode. All HWCFG pins are latched on internal reset release additionally (between 100us – 180us after warm reset assertion) and the status is stored redundantly in STSTAT register by SCU.

- HWCFG[6] and HWCFG[1,2] are recognized as high when the respective pin is open or pulled up to VEXT supply with pull device  $> 2\text{ k}\Omega$  and  $< 4.7\text{ k}\Omega$  on the external system.
- HWCFG[6] and HWCFG[1,2] are recognized as low when the respective pin is pulled down to GND with pull device  $> 2\text{ k}\Omega$  and  $< 4.7\text{ k}\Omega$  on the external system.

The lower limit of the pull resistance is derived from the overload and short specification (see data sheet) in case of a short event.

Regardless of the HWCFG[6] setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate after standby mode entry.

HWCFG [1] P14.5	HWCFG [2] P14.2	HWCFG [3] P14.3	HWCFG [4] P10.5	HWCFG [5] P10.6	HWCFG [6] P14.4
0 - EVR33OFF 1 - EVR33ON	0 - EVRCOFF 1 - EVRCON	0 - Boot from pins HWCFG [5:4] 1 - Flash BMI boot	HWCFG [4:5] [0 0]- Generic Bootstrap (P14.0/1) [0 1]- ABM, Generic Bootstrap on fail (P14.0/1) [1 0]- ABM, ASC Bootstrap on fail (P15.2/3)	[1 1]- Internal start from Flash (weak pull-up active on reset)	Default Pad state 0 - Pins in tristate 1 - Pins with pull-up (weak pull-up active on reset)

1.) HWCFG [1:6] has weak internal pull-up active at start-up if the pin is left unconnected.

**Figure 93 Hardware Configuration (HWCFG) pins**

**Power Management System (PMS)**
**Table 285 Supply Mode and Topology selection**

No.	HWCFG [2,1] <sup>1)</sup>	VGATE1P <sup>2)</sup> VGATE1N <sup>3)</sup>	Supply Pin Voltage Level / Source <sup>4)</sup>	Selected Supply Scheme
a.)	11 <sub>B</sub>	VGATE1P/ VGATE1N connected to gate of P- /N-ch. MOSFET.	VEXT & VEVRSB = 5 V external supply. VDDM = VAREFx = 5V or 3.3V external supply. VFLEX/VFLEX2 = 5 V or 3.3 V. VDDP3 and VDDFL3 supplied by EVR33. VDD supplied by EVRC.	5 V single source supply. EVRC in SMPS mode. EVR33 in LDO mode. 5 V or 3.3 V ADC domain. 5 V or 3.3 V Flexport domain. Standby Mode supported.
d.)	01 <sub>B</sub>	Overlapped P32.1 / P32.0 port pins may be used as standard GPIO.	VEXT & VEVRSB = 5 V external supply. VDDM = VAREFx = 5V or 3.3V external supply. VFLEX/VLFEX2 = 5 V or 3.3 V. VDDP3 and VDDFL3 supplied by EVR33. VDD = 1.25 V external supply.	5 V & 1.25 V external supply. EVRC inactive. EVR33 in LDO mode. 5 V or 3.3 V ADC domain. 5 V or 3.3 V Flexport domain. Standby Mode is supported and 1.25V supply shall be switched off by external regulator after Standby state is entered.
e.)	10 <sub>B</sub>	VGATE1P/ VGATE1N connected to gate of P- /N-ch. MOSFET.	VEXT,VEVRSB,VDDP3, VFLEX/VFLEX2 and VDDFL3 = 3.3V external supply. VDDM = VAREFx = 5V or 3.3V external supply. VDD supplied by EVRC.	3.3 V single source supply. EVRC in SMPS mode. EVR33 inactive. 5 V or 3.3 V ADC domain. 3.3 V Flexport domain. Standby Mode supported.
			VEXT & VEVRSB = 5 V external supply. VDDP3, VFLEX/VFLEX2 and VDDFL3 = 3.3V external supply. VDDM = VAREFx = 5V or 3.3V external supply. VDD supplied by EVRC.	5 V & 3.3 V external supply. EVRC in SMPS mode. EVR33 inactive. 5 V or 3.3 V ADC domain. 5 V or 3.3 V Flexport domain. Standby Mode is supported and 3.3V supply shall be switched off by external regulator after Standby state is entered.
h.)	00 <sub>B</sub>	Overlapped P32.1 / P32.0 port pins may be used as standard GPIO.	VEXT & VEVRSB = 5 V external supply. VDDM = VAREFx = 5V or 3.3V external supply. VFLEX/VFLEX2 = 5 V or 3.3 V external supply. VDDP3 and VDDFL3 = 3.3V external supply. VDD = 1.25 V external supply.	5 V, 3.3 V and 1.25 V are supplied externally. EVRC and EVR33 inactive. 5 V or 3.3 V ADC domain. 5 V or 3.3 V Flexport domain. Standby Mode is supported and 3.3V and 1.25V supplies shall be switched off by external regulator after Standby state is entered.

1) if HWCFG[2,1] pins are left unconnected, it is ensured that EVR33 and EVRC are active owing to the internal weak pull-up active by default after start-up/cold PORST.

## Power Management System (PMS)

- 2) VGATE1P pin is connected to a P- ch. MOSFET in case of EVRC. In case EVRC is inactive via HWCFG2 = 0, VGATE1P pin behaves like a normal port pin (P32.1) and is by default configured as input with weak internal pull-up after start-up/cold PORST.
- 3) VGATE1N pin needs to be connected to the gate of the N- channel MOSFET in case EVRC regulator mode is selected. In case EVRC is inactive via HWCFG2 = 0, VGATE1N pin behaves like a normal port pin (P32.0) and is default configured as input with weak internal pull-up after start-up/cold PORST.
- 4) Only Nominal supply voltage values of respective rails are indicated in the table. The tolerances of the supply voltages are documented in datasheet.

**Table 286 5 V Nominal Supply : Voltage variations at independent supply rails during system modes**

Voltage Rail	5 V Start-up till cold PORST release	5 V Operation RUN mode SLEEP mode	5 V Cranking	5 V VEVRSB STANDBY mode	5 V (VEVRSB + VEXT) STANDBY mode	5 V ED STANDBY mode
$V_{EVRSB}$	2.6 - 5.5 EVRx Start-up	4.5 - 5.5	2.97 - 5.5	2.6 - 5.5	2.97 - 5.5	0
$V_{EXT}$	2.6 - 5.5 EVRx Start-up	4.5 - 5.5	2.97 - 5.5	0	2.97 - 5.5	
$V_{FLEX}/V_{FLEX2}^1)$	Supplied modules in reset	2.97 - 3.63 4.5 - 5.5	2.97 - 5.5		2.97 - 5.5 0 V	
$V_{EBU}$		2.97 - 3.63 4.5 - 5.5	2.97 - 5.5		2.97 - 5.5 0 V	
$V_{DDM}$		2.97 - 3.8 3.8 - 5.5	2.97 - 5.5		2.97 - 5.5 0 V	
$V_{DDP3}$	Supply Ramp-up	2.97 - 3.63	2.6 - 3.63 <sup>2)</sup>		0 V	
$V_{DD}$	Phase. Supplied modules in reset	1.125 - 1.375	1.125 - 1.375			1.0 <sup>3)</sup> - 1.375
$V_{DDSB}$						
$V_{DDPD}^4)$	1.15 - 1.43	1.15 - 1.43	1.15 - 1.43	1.15 - 1.43	1.15 - 1.43	0

- 1) VFLEX2 rail only available on TC37xEXT.
- 2) If EVR33 is used, a minimum VEXT voltage is required to account for pass device drop as documented in datasheet PMS EVR33 section. The voltage is allowed to drop to 2.6V after Flash is set cranking mode where only reading from Flash is allowed with increased wait states.
- 3) 1.0 V permitted at VDDSB only for ED RAM data retention mode as documented in Emulation device section.
- 4) Supply level at internal VDDPD pad

## Power Management System (PMS)

**Table 287 3.3 V Nominal Supply : Voltage variations at independent supply rails during system modes**

Voltage Rail	3.3 V Start-up till cold PORST release	3.3 V Operation RUN mode SLEEP mode	3.3 V Cranking	3.3 V VEVRSB STANDBY mode	3.3 V (VEVRSB + VEXT) STANDBY mode	3.3 V ED STANDBY mode
$V_{EVRSB}$	2.6 - 3.63 EVRx Start-up	2.97 - 3.63	2.97 - 3.63	2.6 - 3.63	2.97 - 3.63	0
$V_{EXT}$	2.6 - 3.63 EVRx Start-up	2.97 - 3.63	2.97 - 3.63	0	2.97 - 3.63	
$V_{FLEX}/V_{FLEX2}^1)$	Supplied modules in reset	2.97 - 3.63	2.97 - 3.63		2.97 - 3.63	
$V_{EBU}$		2.97 - 3.63	2.97 - 3.63		2.97 - 3.63	
$V_{DDM}$		2.97 - 3.8 3.8 - 5.5	2.97 - 5.5		2.97 - 3.63	
$V_{DDP3}$	Supply Ramp-up Phase. Supplied modules in reset	2.97 - 3.63	2.97 - 3.63		0 V	
$V_{DD}$		1.125 - 1.375	1.125 - 1.375			1.0 - 1.375
$V_{DDSB}$						
$V_{DDPD}$	1.15 - 1.43	1.15 - 1.43	1.15 - 1.43	1.15 - 1.43	1.15 - 1.43	0

1) VFLEX2 rail only available on TC37xEXT.

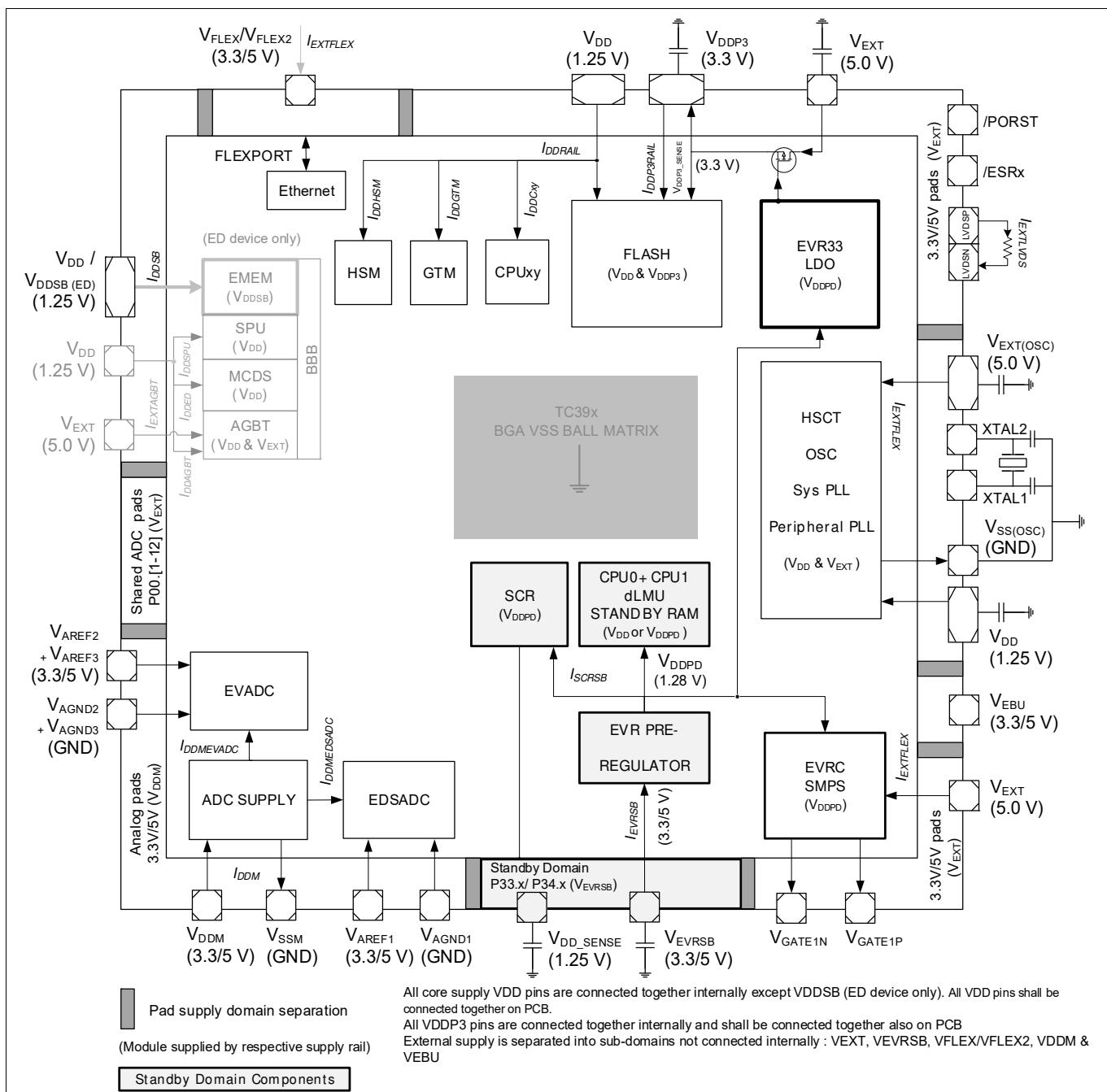
**Table 288 Allowed Combinations of Nominal External Supply Voltages between Voltage Rails<sup>1)</sup>**

Supply Rails	VEXT = VEVRSB = 5V Nominal Voltage Level				VEXT = VEVRSB = 3.3V Nominal Voltage Level			
$V_{EVRSB}$	5 V <sup>2)</sup>				3.3 V			
$V_{EXT}$	5 V				3.3 V			
$V_{FLEX}/V_{FLEX2}^3)$	5 V	3.3 V	5 V	3.3 V	3.3 V			
$V_{EBU}^4)$	5 V		3.3 V		3.3 V			
$V_{DDM}^5)$	5 V				5 V <sup>6)</sup>	3.3 V		
$V_{DDP3}^7)$	3.3 V (external supply or generated by EVR33 )				3.3 V (external supply or generated by EVR33 )			
$V_{DD}$	1.25 V (external supply or generated by EVRC )				1.25 V (external supply or generated by EVRC )			
$V_{DDSB}^8)$	1.25 V (external supply or generated by EVRC )				1.25 V (external supply or generated by EVRC )			

- 1) All supply rails shall have ramped up to their minimum voltage operational limits as documented in the datasheet before warm PORST reset release. It is not allowed to leave any supply rail unsupplied after warm PORST reset release.
- 2) VEVRSB supply rail can be ramped down during VEVRSB Standby mode to 2.6 V minimum voltage.
- 3) VFLEX/VFLEX2 supply rails provide supply to specific ports and can be supplied with nominal 3.3V supply when remaining ports are supplied with nominal 5V. VFLEX /VFLEX2 may be supplied by the same external supply source connected also to VEXT supply rail. VFLEX/VFLEX2 supply level shall be less than or equal to VEXT supply level. VFLEX2 rail only available on TC37xEXT.
- 4) VEBU supply rail provides supply to ports P24, P25, P26, P30 and P31 and can be supplied with nominal 3.3V supply when remaining ports are supplied with nominal 5V. VEBU maybe supplied by the same external supply source connected also to VEXT supply rail. VEBU supply level shall be less than or equal to VEXT supply level.

## Power Management System (PMS)

- 5) VDDM analog supply and VAREFx analog reference supply shall have the same supply level. It is recommended to supply VDDM and VAREFx from the same external supply source with filters.
- 6) VDDM supplies only a part of analog pins. For shared analog pins supplied by VEXT (P00) and VEVRSB (P33), the voltage levels of the respective analog channels would be bounded by the respective supply voltages when they are lower than the VDDM / VAREF voltages.
- 7) EVR33 is designed to supply the current required only by VDDP3 rail and the associated modules requiring 3.3V supply. It is not intended to supply VFLEX/VFLEX2 and VEBU pad currents from 3.3V VDDP3 rail when EVR33 generates VDDP3 supply.
- 8) VDDSB shall be connected to VDD rail and supplied together in case of non emulation devices.



**Figure 94 TC39x Supply Pins and Module Connectivity**

## Power Management System (PMS)

### 11.2.1.2 Supply Ramp-up and Ramp-down Behavior

#### 11.2.1.2.1 Single Supply mode (a)

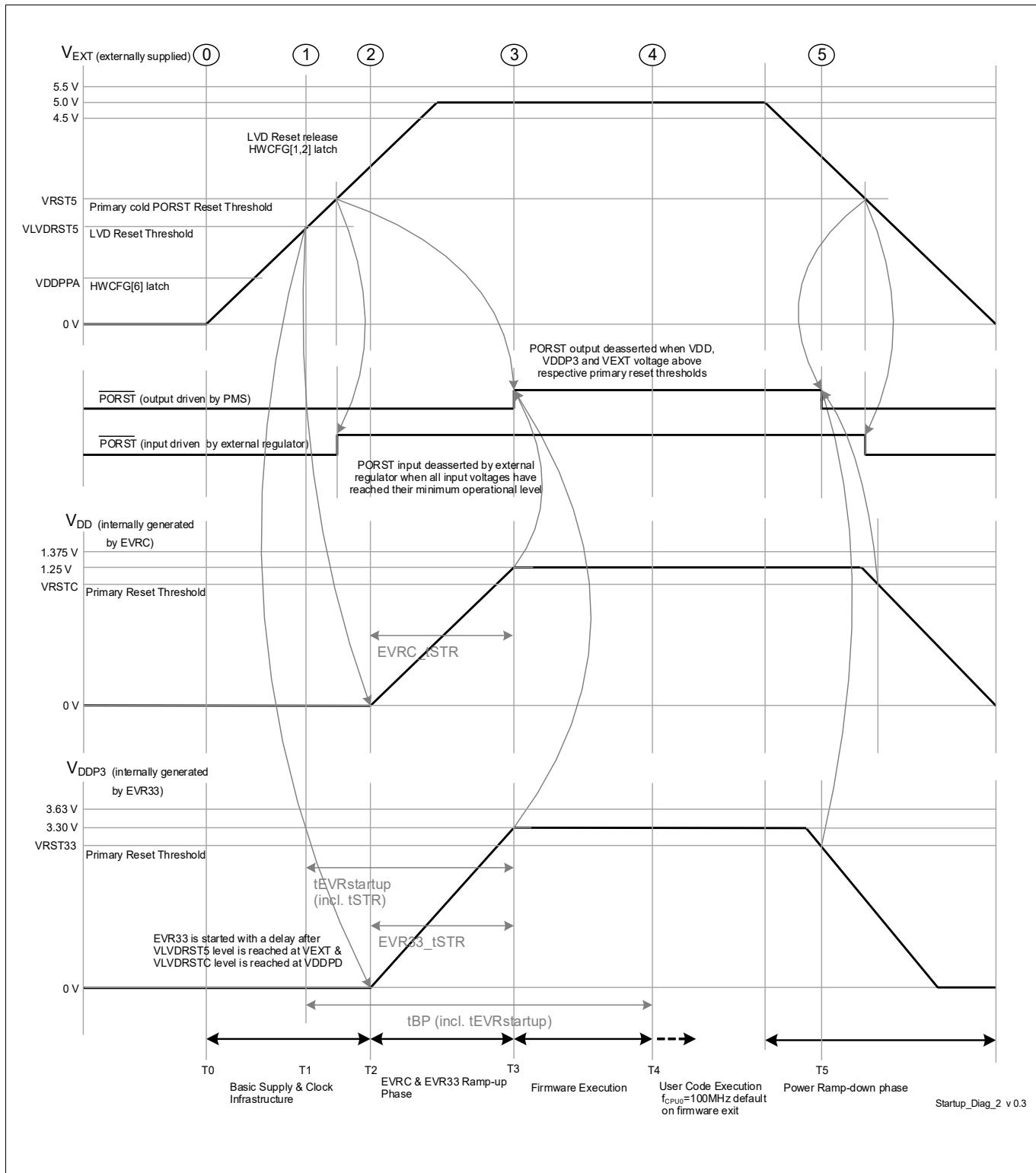


Figure 95 Single Supply mode (a) - VEXT (5 V) single supply

## Power Management System (PMS)

VEXT = 5 V single supply mode. VDD and VDDP3 are generated internally by the EVRC and EVR33 internal regulators.

- The rate at which current is drawn from the external regulator ( $dI_{EXT}/dt$ ) is limited during the basic infrastructure and EVRx regulator start-up phase (T0 up to T2) to a maximum of 100 mA with 100  $\mu$ s settling time. Start-up slew rates for supply rails shall comply to datasheet parameter SR. The slope is defined as the maximal tangential slope between 0% to 100% voltage level. Actual waveform may not represent the specification.
- Furthermore it is also ensured that the current drawn from the regulator ( $dI_{DD}/dt$ ) is limited during the Firmware start-up phase (T3 up to T4) to a maximum of 100 mA with 100  $\mu$ s settling time.
- PORST is active/asserted when either PORST (input) or PORST (output) is active/asserted.
- PORST (input) active means that the reset is held active by external agents by pulling the PORST pin low. It is recommended to keep the PORST (input) asserted until the external supply is above the respective primary reset threshold.
- PORST (output) active means that  $\mu$ C asserts the reset internally and drives the PORST pin low thus propagating the reset to external devices. The PORST (output) is asserted by the  $\mu$ C when atleast one among the three supply domains (VDD, VDDP3 or VEXT) violate their primary under-voltage reset thresholds. The PORST (output) is de-asserted by the  $\mu$ C when all supplies are above their primary reset thresholds and the basic supply and clock infrastructure is available. During reset release at T3, the load jump of upto 150 mA ( $dI_{DD}$ ) is expected.
- The power sequence as shown in [Figure 95](#) is enumerated below
  - T1 up to T2 refers to the period in time when basic supply and clock infrastructure components are available as the external supply ramps up. The bandgap and internal clock sources are started .The supply mode is evaluated based on the HWCFG[2:1,6] pins. These events are initiated after LVD reset release at T1. LVD reset is released when both input voltages VEXT and VEVRSB are above VLVRST5 and VLVRSTSB levels respectively. Internal pre-regulator VDDPD voltage is above VLVRSTC level.
  - T2 refers to the point in time where consequently a soft start of EVRC and EVR33 regulators are initiated. PORST (input) does not have any affect on EVR33 or EVRC output and regulators continue to generate the respective voltages though PORST is asserted and the device is in reset state. The generated voltage follows a soft ramp-up over the tSTR (datasheet parameter) time to avoid overshoots.
  - T3 refers to the point in time when all supplies are above their primary reset thresholds denoted by VRST5, VRST33 and VRSTC supply voltage levels. EVRC and EVR33 regulators have ramped up. PORST (output) is de-asserted and HWCFG[3:5] pins are latched on PORST rising edge by SCU. Firmware execution is initiated. The time between T1 and T3 is documented as tEVRstartup (datasheet parameter).
  - T4 refers to the point in time when Firmware execution is completed and User code execution starts with CPU0 at a default frequency of 100 MHz. The time between T0 and T4 is documented as tBP (datasheet parameter).
  - T5 refers to the point in time during the ramp-down phase when at least one of the externally provided or generated supplies (VDD, VDDP3 or VEXT) drop below their respective primary under-voltage reset thresholds.

## Power Management System (PMS)

### 11.2.1.2.2 Single Supply mode (e)

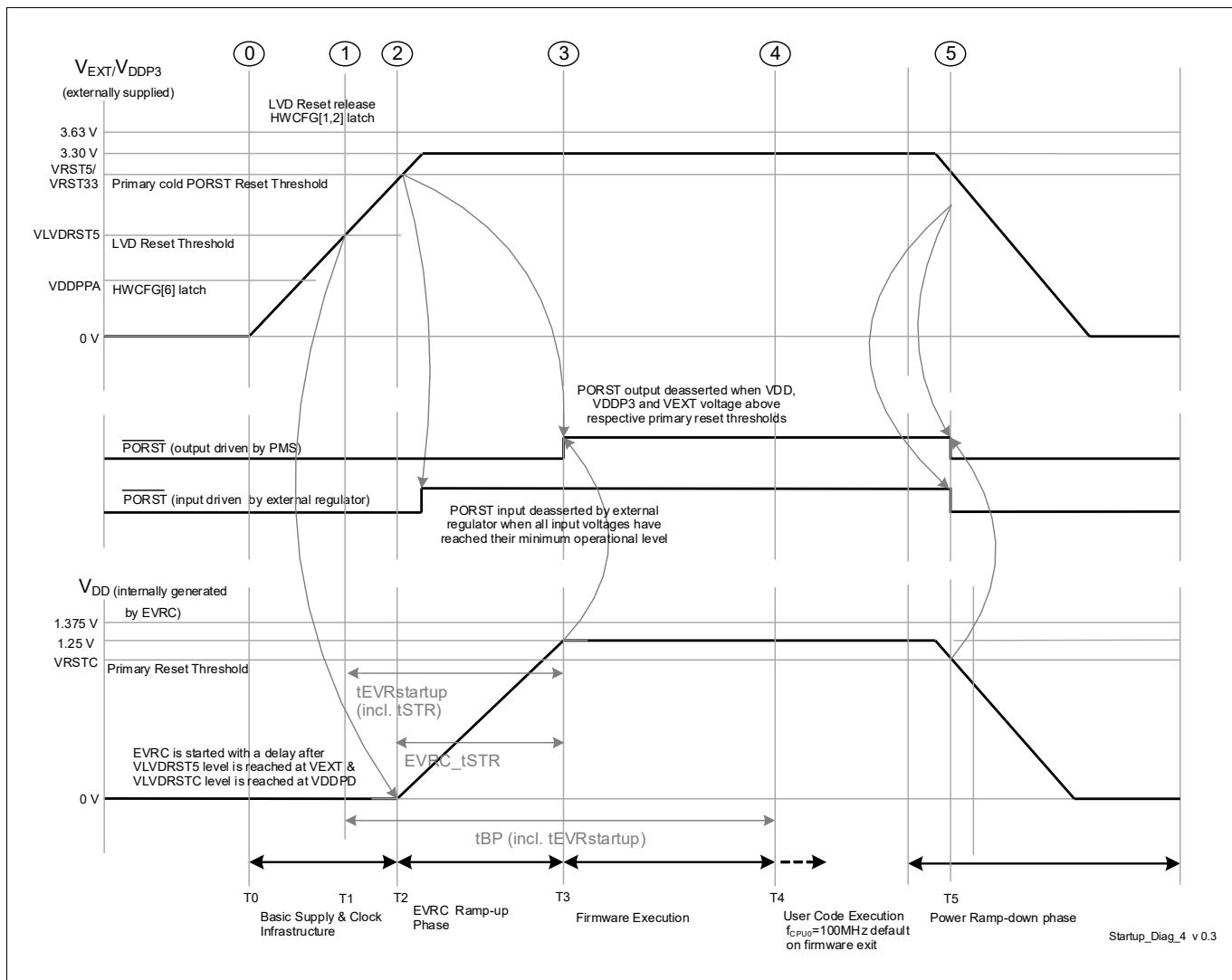


Figure 96 Single Supply mode (e) - (VEXT & VDDP3) 3.3 V single supply

V<sub>EXT</sub> = V<sub>DDP3</sub> = 3.3 V single supply mode. VDD is generated internally by the EVRC regulator.

- The rate at which current is drawn from the external regulator ( $dI_{EXT}/dt$ ) is limited in the Start-up phase to a maximum of 100 mA with 100  $\mu$ s settling time. Start-up slew rates for supply rails shall comply to datasheet parameter SR. The slope is defined as the maximal tangential slope between 0% to 100% voltage level. Actual waveform may not represent the specification.
- PORST is active/asserted when either PORST (input) or PORST (output) is active/asserted.
- PORST (input) active means that the reset is held active by external agents by pulling the PORST pin low. It is recommended to keep the PORST (input) asserted until the external supply is above the respective primary reset threshold.
- PORST (output) active means that  $\mu$ C asserts the reset internally and drives the PORST pin low thus propagating the reset to external devices. The PORST (output) is asserted by the  $\mu$ C when at least one among the three supply domains (VDD, VDDP3 or VEXT) violate their primary under-voltage reset thresholds. The PORST (output) is de-asserted by the  $\mu$ C when all supplies are above their primary reset thresholds and the

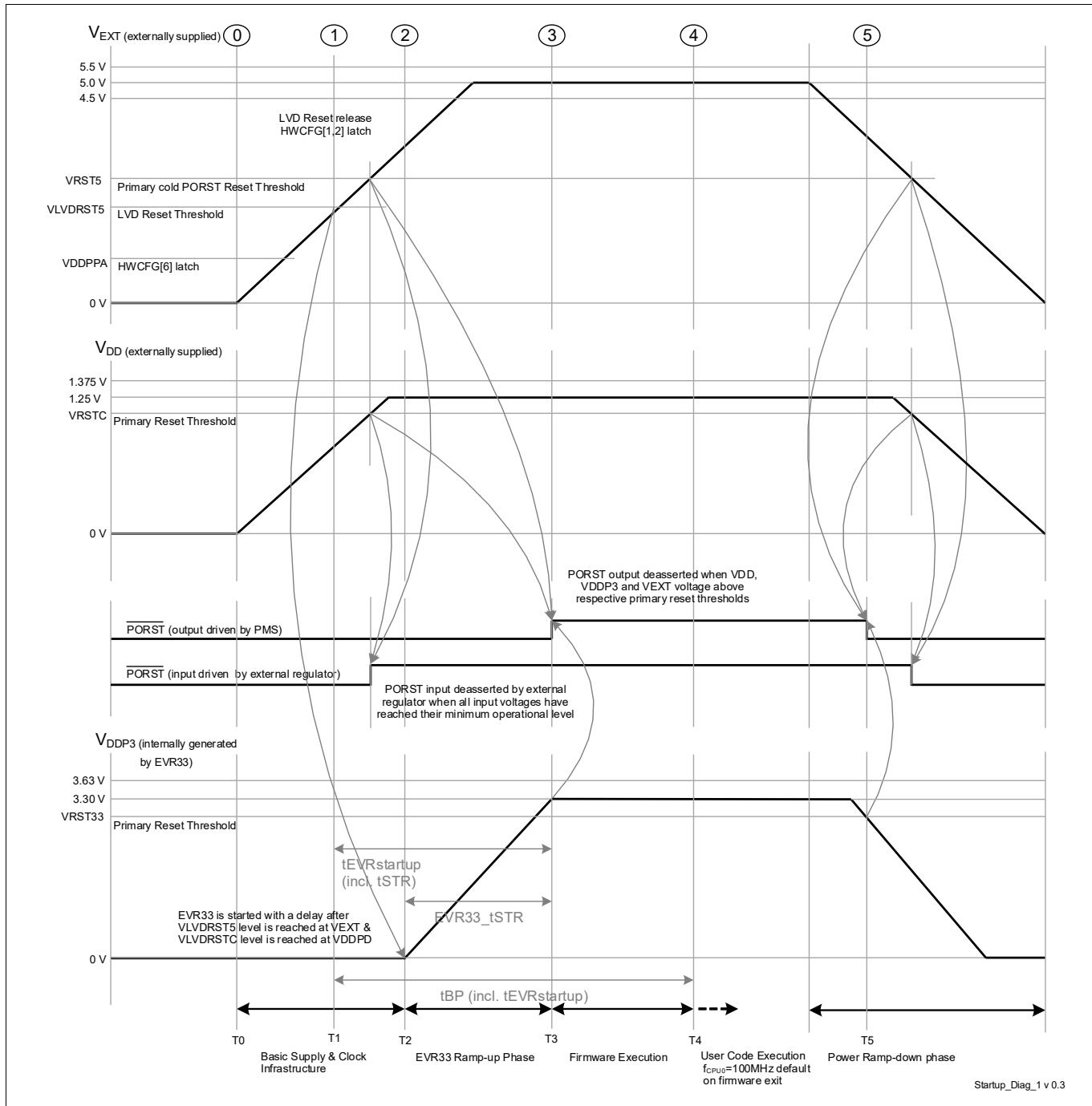
## Power Management System (PMS)

basic supply and clock infrastructure is available. During reset release at T3, the load jump of upto 150 mA (dIDD) is expected.

- The power sequence as shown in [Figure 96](#) is enumerated below
  - T1 up to T2 refers to the period in time when basic supply and clock infrastructure components are available as the external supply ramps up. The bandgap and internal clock sources are started .The supply mode is evaluated based on the HWCFG[2:1,6] pins. These events are initiated after LVD reset release at T1. LVD reset is released when both input voltages VEXT and VEVRSB are above VLVRST5 and VLVRSTSB levels respectively. Internal pre-regulator VDDPD voltage is above VLVRSTC level.
  - T2 refers to the point in time where consequently a soft start of EVRC regulator is initiated. PORST (input) does not have any affect on EVRC output and regulators continue to generate the respective voltages though PORST is asserted and the device is in reset state. The generated voltage follows a soft ramp-up over the tSTR (datasheet parameter) time to avoid overshoots.
  - T3 refers to the point in time when all supplies are above their primary reset thresholds denoted by VRST5, VRST33 and VRSTC supply voltage levels. EVRC regulator has ramped up. PORST (output) is de-asserted and HWCFG[3:5] pins are latched on PORST rising edge by SCU. Firmware execution is initiated. The time between T1 and T3 is documented as tEVStartup (datasheet parameter).
  - T4 refers to the point in time when Firmware execution is completed and User code execution starts with CPU0 at a default frequency of 100 MHz. The time between T0 and T4 is documented as tBP (datasheet parameter).
  - T5 refers to the point in time during the ramp-down phase when at least one of the externally provided or generated supplies (VDD, VDDP3 or VEXT) drop below their respective primary under-voltage reset thresholds.

## Power Management System (PMS)

### 11.2.1.2.3 External Supply mode (d)



**Figure 97 External Supply mode (d) - VEXT and VDD externally supplied**

V<sub>EXT</sub> = 5 V and V<sub>DD</sub> supplies are externally supplied. 3.3V is generated internally by the EVR33 regulator.

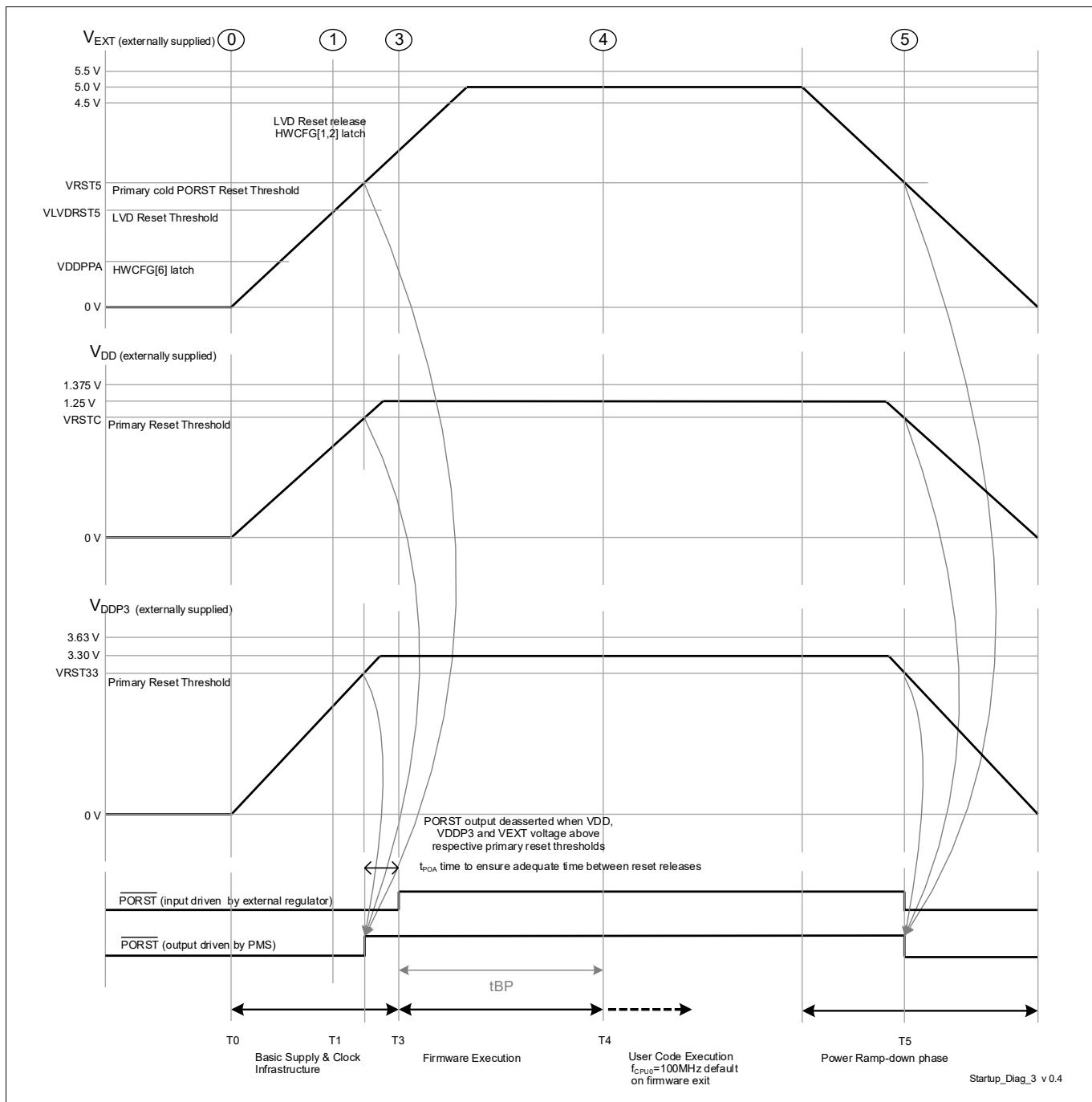
- External supplies V<sub>EXT</sub> and V<sub>DD</sub> may ramp-up or ramp-down independent of each other with regards to start, rise and fall time(s). Start-up slew rates for supply rails shall comply to datasheet parameter SR. The slope is defined as the maximal tangential slope between 0% to 100% voltage level. Actual waveform may not represent the specification. It is expected that during start-up, V<sub>EXT</sub> ramps up before V<sub>DD</sub> rail. If V<sub>DD</sub> voltage rail is ramped up before V<sub>EXT</sub>; V<sub>DD</sub> supply overshoots during start-up shall be limited within the operational voltage range.

## Power Management System (PMS)

- The rate at which current is drawn from the external regulator ( $dI_{EXT}/dt$  or  $dI_{DD}/dt$ ) is limited in the Start-up phase to a maximum of 100 mA with 100  $\mu$ s settling time.
- PORST is active/asserted when either PORST (input) or PORST (output) is active/asserted.
- PORST (input) active means that the reset is held active by external agents by pulling the PORST pin low. It is recommended to keep the PORST (input) asserted until all the external supplies are above their primary reset thresholds.
- PORST (output) active means that  $\mu$ C asserts the reset internally and drives the PORST pin low thus propagating the reset to external devices. The PORST (output) is asserted by the  $\mu$ C when at least one among the three supply domains (VDD, VDDP3 or VEXT) violate their primary under-voltage reset thresholds. The PORST (output) is de-asserted by the  $\mu$ C when all supplies are above their primary reset thresholds and the basic supply and clock infrastructure is available. During reset release at T3, the load jump of up to 150 mA ( $dI_{DD}$ ) is expected.
- The power sequence as shown in [Figure 97](#) is enumerated below
  - T1 up to T2 refers to the period in time when basic supply and clock infrastructure components are available as the external supply ramps up. The bandgap and internal clock sources are started. The supply mode is evaluated based on the HWCFG[2:1,6] pins. These events are initiated after LVD reset release at T1. LVD reset is released when both input voltages VEXT and VEVRSB are above VLVRST5 and VLVRSTSB levels respectively. Internal pre-regulator VDDPD voltage is above VLVRSTC level.
  - T2 refers to the point in time where consequently a soft start of EVR33 regulator is initiated. PORST (input) does not have any affect on EVR33 output and regulators continue to generate the respective voltages though PORST is asserted and the device is in reset state. The generated voltage follows a soft ramp-up over the tSTR (datasheet parameter) time to avoid overshoots.
  - T3 refers to the point in time when all supplies are above their primary reset thresholds denoted by VRST5, VRST33 and VRSTC supply voltage levels. EVR33 regulators has ramped up. PORST (output) is de-asserted and HWCFG[3:5] pins are latched on PORST rising edge by SCU. Firmware execution is initiated. The time between T1 and T3 is documented as tEVStartup (datasheet parameter).
  - T4 refers to the point in time when Firmware execution is completed and User code execution starts with CPU0 at a default frequency of 100 MHz. The time between T0 and T4 is documented as tBP (datasheet parameter).
  - T5 refers to the point in time during the ramp-down phase when at least one of the externally provided or generated supplies (VDD, VDDP3 or VEXT) drop below their respective primary under-voltage reset thresholds.

## Power Management System (PMS)

### 11.2.1.2.4 External Supply mode (h)



**Figure 98 External Supply mode (h) - VEXT, VDDP3 & VDD externally supplied**

All supplies, namely VEXT, VDDP3 & VDD are externally supplied.

- External supplies VEXT, VDDP3 & VDD may ramp-up or ramp-down independent of each other with regards to start, rise and fall time(s). Start-up slew rates for supply rails shall comply to datasheet parameter SR. The slope is defined as the maximal tangential slope between 0% to 100% voltage level. Actual waveform may not represent the specification. It is expected that during start-up, VEXT ramps up before VDDP3 and VDD rails. If smaller voltage rails are ramped up before VEXT; VDD and VDDP3 supply overshoots during start-up shall be limited within the operational voltage ranges of the respective rails.

## Power Management System (PMS)

- The rate at which current is drawn from the external regulator ( $dI_{EXT}/dt$ ,  $dI_{DD}/dt$  or  $dI_{DDP3}/dt$ ) is limited in the Start-up phase to a maximum of 100 mA with 100  $\mu$ s settling time.
- PORST is active/asserted when either PORST (input) or PORST (output) is active/asserted.
- PORST (input) active means that the reset is held active by external agents by pulling the PORST pin low. It is recommended to keep the PORST (input) asserted until all the external supplies are above their primary reset thresholds.
- PORST (output) active means that  $\mu$ C asserts the reset internally and drives the PORST pin low thus propagating the reset to external devices. The PORST (output) is asserted by the  $\mu$ C when at least one among the three supply domains (VDD, VDDP3 or VEXT) violate their primary under-voltage reset thresholds. The PORST (output) is de-asserted by the  $\mu$ C when all supplies are above their primary reset thresholds and the basic supply and clock infrastructure is available. During reset release at T3, the load jump of up to 150 mA ( $dI_{DD}$ ) is expected.
- The power sequence as shown in [Figure 98](#) is enumerated below
  - T1 up to T3 refers to the period in time when basic supply and clock infrastructure components are available as the external supply ramps up. The bandgap and internal clock sources are started. The supply mode is evaluated based on the HWCFG[2:1,6] pins. These events are initiated after LVD reset release at T1. LVD reset is released when both input voltages VEXT and VEVRSB are above VLVRST5 and VLVRSTSB levels respectively. Internal pre-regulator VDDPD voltage is above VLVRSTC level.
  - T3 refers to the point in time when all supplies are above their primary reset thresholds denoted by VRST5, VRST33 and VRSTC supply voltage levels. PORST (output) is de-asserted and HWCFG[3:5] pins are latched on PORST rising edge by SCU. Firmware execution is initiated.
  - T4 refers to the point in time when Firmware execution is completed and User code execution starts with CPU0 at a default frequency of 100 MHz. The time between T0 and T4 is documented as tBP (datasheet parameter).
  - T5 refers to the point in time during the ramp-down phase when at least one of the externally provided supplies (VDD, VDDP3 or VEXT) drop below their respective primary under-voltage reset thresholds.

## Power Management System (PMS)

### 11.2.1.2.5 HWCFG, P32.1 / VGATE1P, P32.0 / VGATE1N behavior during Start-up

EVRC SMPS regulator inactive (HWCFG [2] = 0)				
	P32.0 pad	VGATE1N pad	VGATE1P pad	P32.1 pad
VEXT Ramp-up	PU till HWCFG6 latch @ VDDPA. PU or Z based on HWCFG6 latch	Z	PU till HWCFG[2] latch	PU till HWCFG6 latch @ VDDPA. PU or Z based on HWCFG6 latch
RUN	Active		Z	Active

EVRC SMPS regulator active (HWCFG [2] = 1)				
	P32.0 pad	VGATE1N pad	VGATE1P pad	P32.1 pad
VEXT Ramp-up	PU till HWCFG6 latch @ VDDPA. PU or Z based on HWCFG6 latch	Z	PU	PU till HWCFG6 latch @ VDDPA. PU or Z based on HWCFG6 latch
EVRC Ramp-up	Z enps=0, trist=1	Active	Active	Z enps=0, trist=1
RUN		Active	Active	

**Figure 99 VGATE1P and VGATE1N pin connectivity**

VGATE1P / P32.1 pin and VGATE1N / P32.0 pins shall be connected to drive external Pch. and Nch. MOSFET if EVRC is active in SMPS mode. If VDD core supply is provided externally, P32.0 and P32.1 pins may be used as normal GPIOs. Each pin is double bonded or connected to two separate pads, namely VGATEEx pad dedicated for EVRC MOSFET drive function and P32.x pad supporting regular PORT / GPIO functions as shown in [Figure 99](#). The function for the respective pins are defined by HWCFG [2] pin as to whether internal EVRC regulator is used or not.

If EVRC regulator is inactive configured via HWCFG [2] = 0, then GPIO function is available and the pin control via PORT module is allowed. The default reset behavior of the P32.x is dependent on HWCFG[6] level as to whether pull-up or tristate request is active. If pin is used for GPIO function, it should be noted that these pads exhibit higher leakages as documented in the datasheet.

If EVRC regulator is activated via HWCFG [2] = 1, then GPIO function is no more available and the behavior of the pins during start-up is as portrayed in [Figure 100](#). During VEXT ramp-up, the VGATE1P / P32.1 pin is pulled high so that P-ch. MOSFET is completely switched off. Once the HWCFG[2:1] pins are latched, EVRC is ramped up and the regulator drives a PWM with increasing duty cycle via VGATE1P / P32.1 pin. The VGATE1N / P32.0 pin remains in tristate during VEXT and EVRC ramp-up phases respectively. Only after the EVRC VDD voltage output has crossed a minimum threshold does the regulator starts driving the N-ch. MOSFET via VGATE1N / P32.0 pin.

Furthermore in case of power-fail on VEXT or VEVRSB rails, the SMPS regulator is kept active for 4 clock cycles at 100MHz before the regulator goes into reset state . This ensures that the PMOS is turned-off properly so as to avoid overshoots on the VDD rail irrespective of the switching phase.

## Power Management System (PMS)

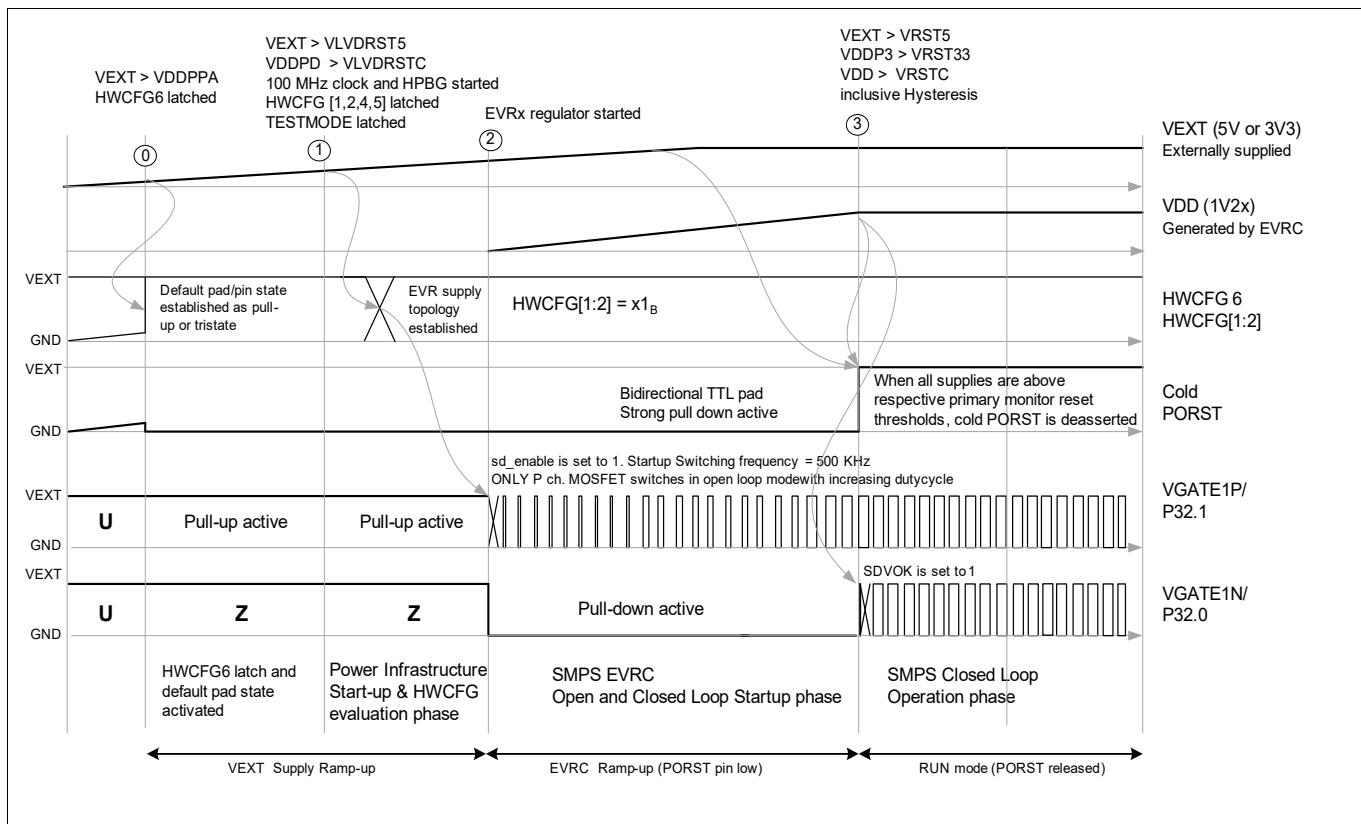


Figure 100 VGATE behavior during start-up when EVRC regulator is used

## Power Management System (PMS)

### 11.2.1.3 PMS Infrastructure Components

Power Management System constitutes infrastructure components which need to be started before ramping the EVR33 & EVRC Embedded Voltage Regulators.

- EVR Pre- Regulator (EVRPR)
- 100 MHz Back-up Clock Source (fBACK)
- Secondary High Precision Bandgap reference (SHPBG)
- 70 kHz Standby Clock Source (fSB)
- Primary Low Power Bandgap reference (PLPBG)

#### 11.2.1.3.1 Independent VEVRSB & VDDPD Supply domain and EVR Pre-Regulator (EVRPR)

The objective of the EVRPR is to supply the basic infrastructure components, the Standby domain and certain safety components with a dedicated low-noise independent supply. The EVRPR pre-regulator is supplied directly by the external 5 V or 3.3 V VEVRSB supply. It is implemented as a low drop-out regulator generating the 1.25 V VDDPD internal voltage which is buffered internally and is not routed to any external supply pin. Since EVRPR part is always powered on as long as the external supply is available and also in Standby mode, it is implemented to have low power consumption to meet IStandby current parameter limits in datasheet. The EVRPR supplies the high precision bandgap, the 100 MHz EVR clock source and EVRC / EVR33 regulators as the regulators have to be independent from their generated supplies. The EVRPR also supplies the Standby domain including the Standby RAMs, the Wake-Up Timer, the Standby Controller and a part of the Port domain (Port 33 / 34).

The minimum power detection logic ensures that a minimum voltage level is available on VEXT and VEVRSB external supplies and on the internally generated VDDPD supply via dedicated detectors. The VEXT supply is monitored for minimum VLDRST5 level to ensure that adequate voltage is available to latch HWCFG pins and start EVRC. Likewise, the internal VDDPD supply is monitored for minimum VLDRSTC voltage level by the VDDPD detector with in-built reference. When both conditions are fulfilled the start-up of the EVRPR has been successfully completed and the EVR Low Voltage Detector reset (LVD reset) is released. The 100 MHz clock and high precision bandgap are consequently started. The HWCFG pins are evaluated to establish the supply mode which needs to be activated. Consequently EVRC and EVR33 are started in parallel in a soft ramp-up to ensure a voltage ramp-up with minimal overshoots. In case both EVRC and EVR33 are activated, a normal start-up is completed when both the regulator outputs are stable and operational. Consequently cold PORST reset is released when VEXT, VDDP3 and VDD voltages ramp-ups are complete and the respective voltages are above their minimum operational limits (VRSTxx / VxxPRIUV).

#### 11.2.1.3.2 Reference Voltage Generation : Secondary Bandgap Reference (SHPBG)

The objective of the Secondary High Precision BandGap and Reference current circuitry is to provide an accurate voltage reference and reference currents to various modules. The reference is used by EVRC, EVR33, supply monitors, ADC modules, XTAL Oscillator, Flash, ADC and LVDS Pads.

The secondary high precision bandgap reference is checked against the primary low power bandgap reference or VDDPD voltage to detect bandgap drifts during start-up phase. This is part of the Power BIST ([Section 11.2.2.5.3](#)) which is carried out only during a supply ramp-up.

#### 11.2.1.3.3 100 MHz Back-up Clock Source (fBACK)

The 100 MHz clock source is a precise back-up on-chip clock used by EVRs, firmware and serves as the main system clock during the Start-up phase. It is further used as an independent clock reference for clock monitoring and can be used as a back-up clock in case of loss of lock or crystal failures. After start-up, the 100 MHz clock source has a higher variance in the order of  $\pm 40\%$  and the clock source is later trimmed by the start-up software.

---

## Power Management System (PMS)

as documented in datasheet. It shall be ensured that the PMS subsystem, boot software / Firmware and the start-up modules are tolerant and functionally robust to this clock variation.

The **EVROSCCTRL** register shall not be modified by the application software, as it is configured by the Start-Up Software in order to trim the back-up oscillator to the specified accuracy limits. Additional compensation for improved accuracy across the temperature range is possible by enabling the dynamic oscillator trimming in the register bits **EVROSCCTRL.OSCTEMPOFFS** and **EVROSCCTRL.OSCTRIMEN**.

---

**Power Management System (PMS)****11.2.1.4 Die Temperature Measurement**

The Die Temperature Sensor (DTS) generates a measurement result that indicates directly the current temperature. The DTS measures the temperature with an accuracy within ( $T_{NL} + T_{CALACC}$ ) parameter limits within the TSR temperature range documented in the datasheet. The result of the measurement is updated periodically in **DTSSTAT.RESULT** register bit field with a resolution less than 1/5th of a degree Kelvin. The Die Temperature Sensor is available after cold PORST reset release on a device start-up and temperature measurements are carried out continuously during normal RUN / SLEEP modes. The DTS and corresponding registers are not affected by a warm PORST, system or application reset; consequently DTSTAT temperature result from earlier conversion is available for immediate use after any warm reset.

After an ongoing temperature measurement is completed, **DTSSTAT.RESULT** bit field is updated coherently with the new value. An interrupt service request (SRC\_PMSDTS) can be generated after a measurement is completed. The DTS accuracy and measurement time is defined in the Data Sheet.

Die temperature upper and lower limits are configured in **DTSLIM.UPPER** and **LOWER** register bits. On violation of these limits, **DTSLIM.UOF** and **LLU** status bits are set and alarms are forwarded to SMU and HSM. After start-up, the DTS limits have to re-configured appropriately depending on the application before alarm reactions from SMU or HSM are activated. Only when a new DTS conversion result is available, the DTS comparators are consequently triggered to check the actual **DTSSTAT.RESULT** against the upper and lower limits.

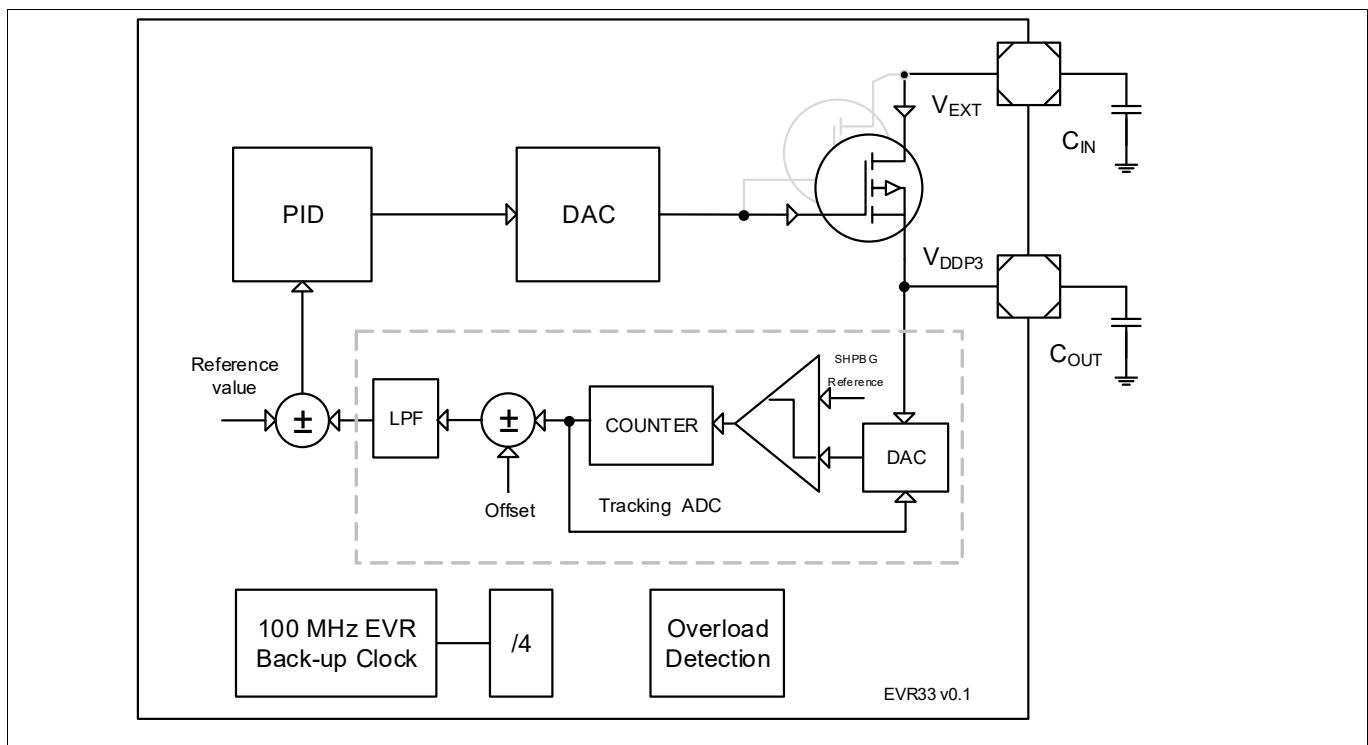
Note: LDMST or SWAPMSK.W should be used only with bit mask enabled for all 'rwh' bits in the **DTSLIM** register.

## Power Management System (PMS)

### 11.2.2 Power Supply Generation and Monitoring

#### 11.2.2.1 Linear Regulator Mode (EVR33)

The EVR33 regulator supplies the Flash module. EVR33 constitutes a digital regulator, a pass device control unit and a voltage feedback loop. In order to compensate technology and process variations, the ADC and the DAC are device individually trimmed. The EVR33 regulator output voltage ( $V_{DDP3}$ ) is measured by a dedicated ADC using SHPBG reference supply and the result is indicated also in register **EVRADCSTAT.ADC33V**. The closed loop regulation cycle is triggered at the end of the ADC conversion. The error difference is fed to a PID controller and the output of the controller is fed to the DAC to control the gate voltage of the pass devices. The pass device outputs are buffered by external capacitor to handle load transients so as not to violate the operating voltage limits. EVR33 can be individually disabled via HWCFG[1] pin as described in [Chapter 11.2.1.1](#). During the Start-up phase, the setpoint voltage is ramped up in steps over the start-up period to ensure a soft ramp-up of the  $V_{DDP3}$  voltage.

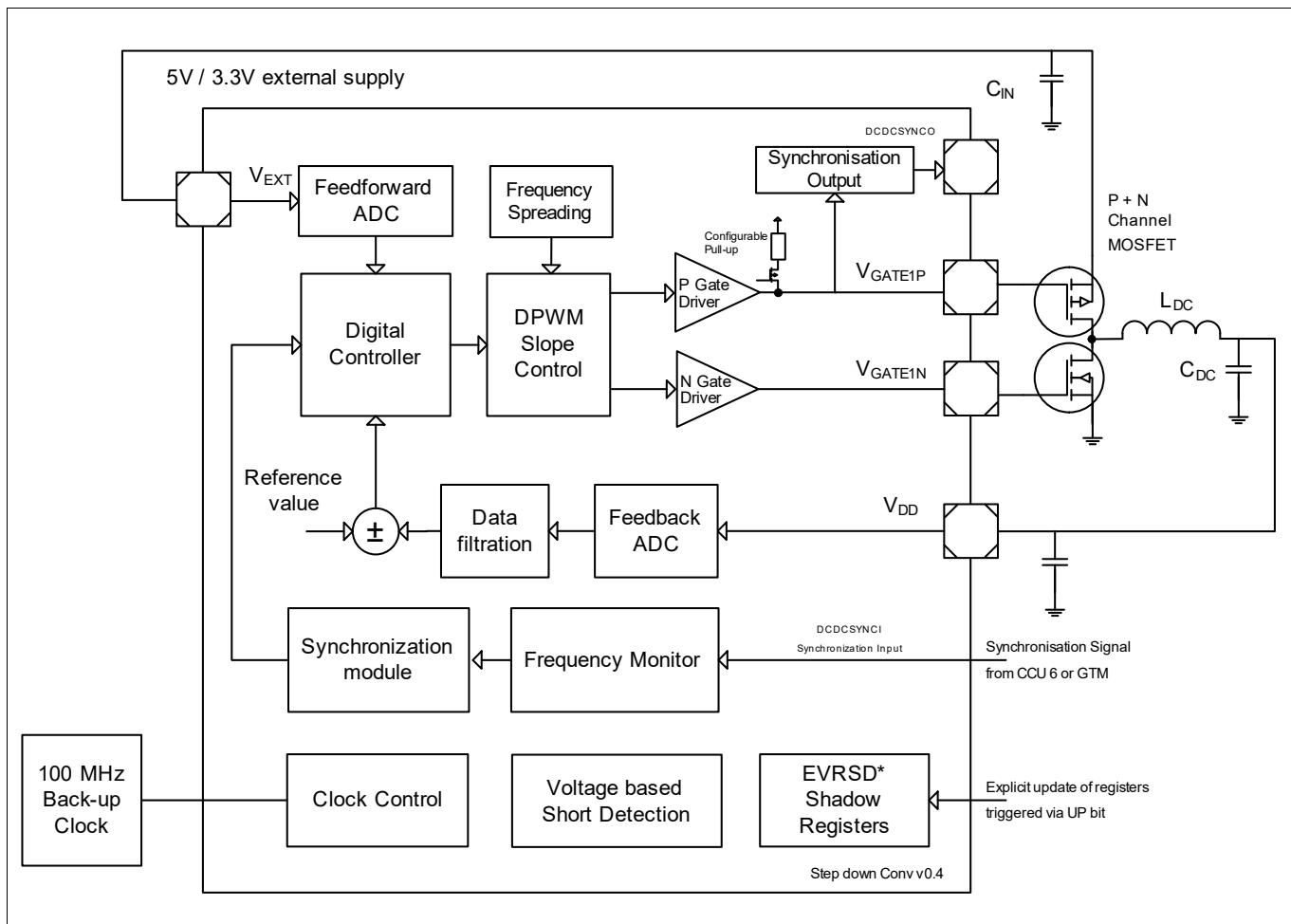


**Figure 101 EVR33 LDO regulator**

## Power Management System (PMS)

### 11.2.2.2 Step-down Regulator (EVRC)

The Step-down regulator provides a higher efficiency of power conversion compared to linear voltage regulator. However it requires additional external components and injects more switching noise into the system. The integrated EVRC regulator modulates an external charge device to buffer the energy in a LC filter in order to generate a regulated core supply. The 5 V or 3.3 V external VEXT supply shall be provided to the complementary MOSFET switches as shown in [Figure 103](#) and [Figure 105](#).



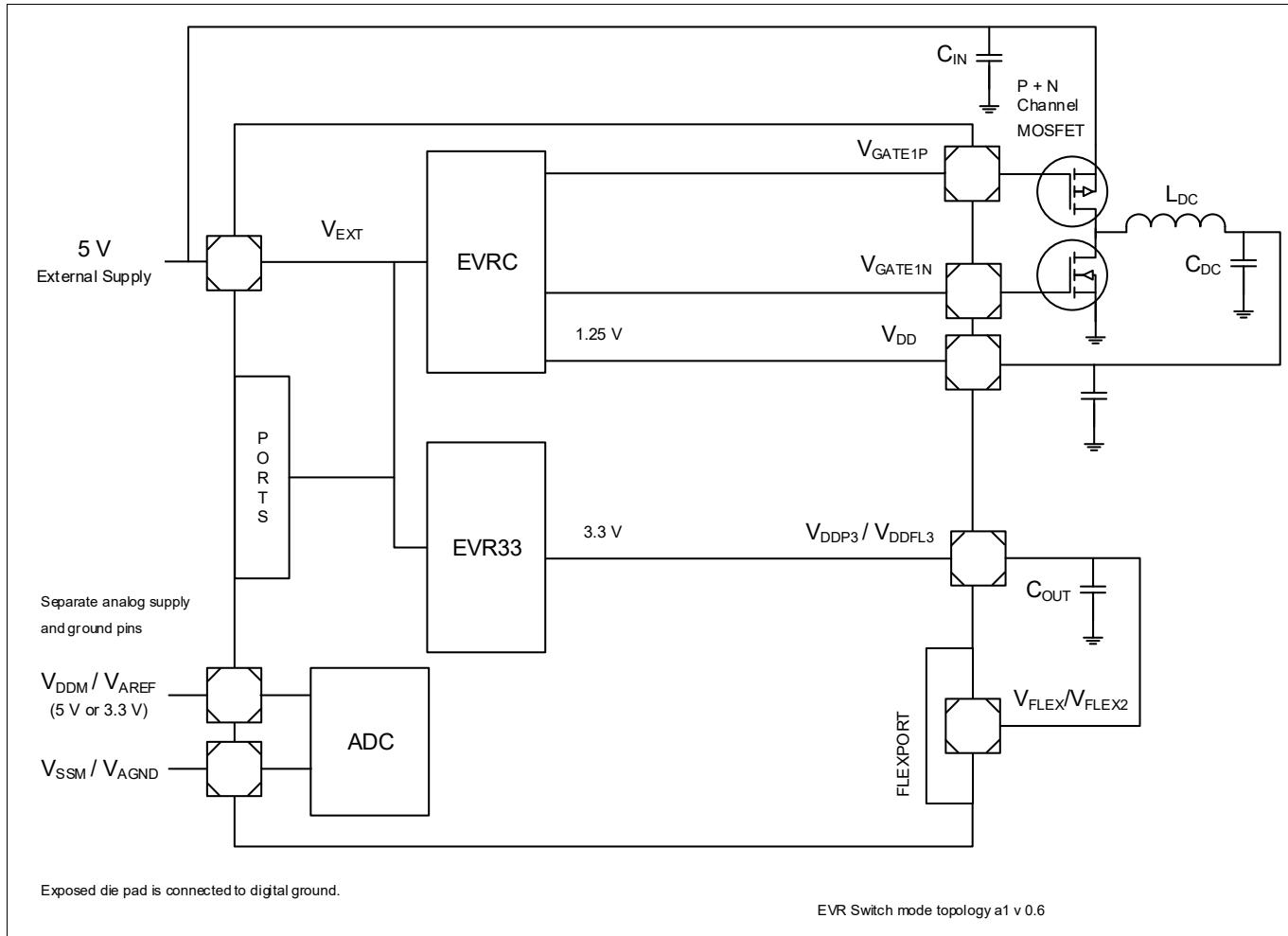
**Figure 102** EVRC Step down regulator

The control strategy involves synchronous switching of the complementary Pch. and Nch. MOSFETs at a defined switching frequency using pulse width modulation. The recommended nominal switching frequency is 1.8 MHz in PWM mode and is derived based on efficiency, performance and EMI/EMC trade-off. The duty cycle has a resolution of the base clock frequency of 100 MHz internal clock and is controlled using a programmable digital controller which reacts to the deviation of the input and output voltage against a reference voltage. The EVRC regulator operates in continuous conductance mode during normal PWM mode. The output voltage value is approximately equal to the input voltage multiplied by the duty cycle factor neglecting the parasitic components and losses.

The output voltage V<sub>DD</sub> is measured with a feedback ADC (FBADC) and sampled with an offset to the PWM switching event to mitigate switching noise influence. The measured output voltage is then fed into the digital filter and provided to the digital controller. The measured core voltage is indicated in [EVRSDSTAT0.ADCFBCV](#) status bits. The target of the digital controller is to compute the new duty cycle and the skip pulse information for the switching period based on the input information. The duty cycle is limited between maximum and minimum value in order to ensure proper commutation in the power switches. The duty cycle is realized by a Digital Pulse

## Power Management System (PMS)

Width Modulator which drives the respective Gate drivers. The duty cycle of the previous period is indicated in **EVRSDSTAT0.DPWMOUT** status bits. In case the controller output is above certain threshold during the start of a switching period and based on internal controller states, a pulse skipping mechanism is employed and the power switch will remain for a whole switching period in OFF state. The parameters for the digital controller are programmable. The external VEXT supply is also measured by the Primary SWD / VEXT Monitor ADC to facilitate a parameter switch if the voltage crosses a threshold and to differentiate between 5 V or 3.3 V external supply case.



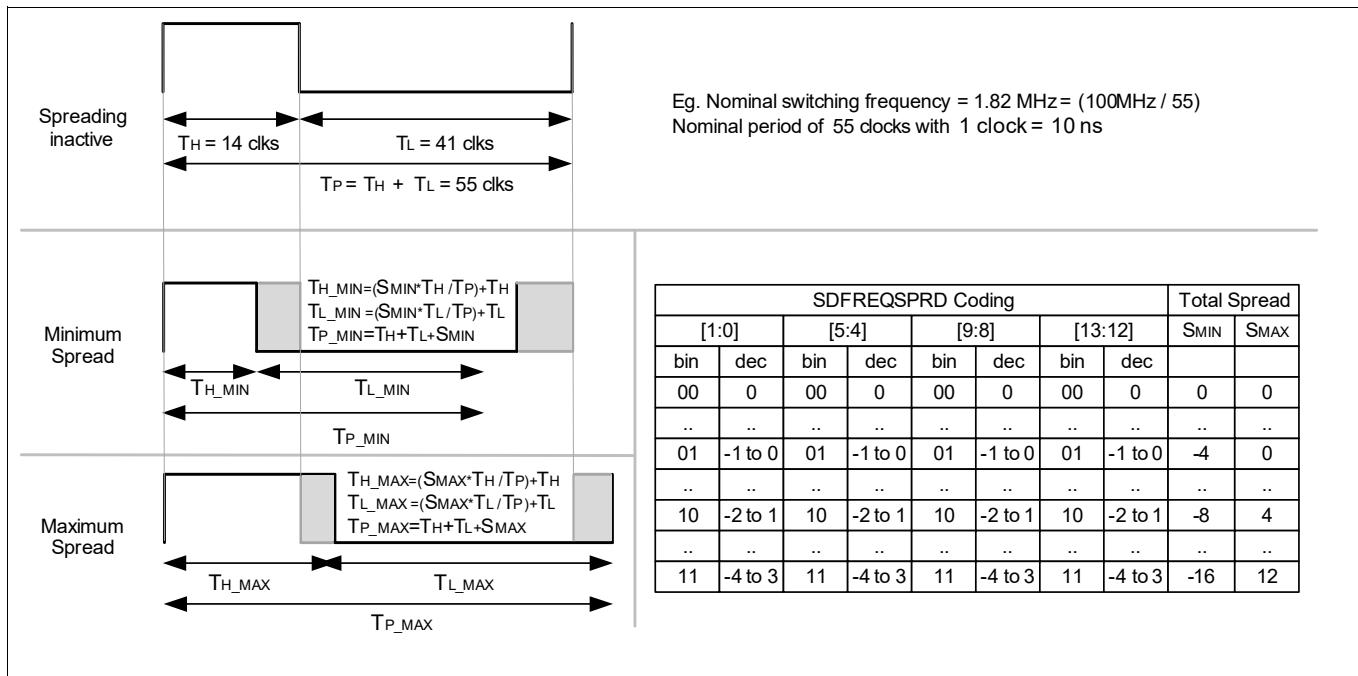
**Figure 103 EVR Switch mode topology (a) - 5 V single supply**

The nominal switching frequency is configured via **EVRSDCTRL0.SDFREQ** register bits with a nominal 10 ns switching period resolution at a base frequency of 100 MHz.

Frequency spreading may be activated via **EVRSDCTRL0.SDFREQSPRD** register bits in case of SMPS mode to comply with EMI / EMC requirements. The actual switching period is varied in a random manner between a minimum and maximum period spread as depicted in **Figure 104**. The maximum spread is bounded by the programmed value in SDFREQSPRD register field. It is ensured that duty cycle changes are compensated during frequency spreading over consecutive periods to ensure that the induced voltage ripple owing to frequency spreading is kept to a minimum. Furthermore, programmable slope control of the external MOSFETs is implemented by controlling the driver strength and slew rate and shall be adapted to the external components. A scaled switching (DCDCSYNCO) output derived from the actual EVRC switching output is routed to an external pin. The frequency of the scaled output is in the range between 50 kHz and the actual EVRC switching frequency. The scaling factor is configured in **EVRSDCTRL7.SYNCDIVFAC** bitfield and the output is enabled via **PMSWCR5.DCDCSYNCO** bitfield. The EVRC switching output with the current dutycycle is routed directly to the pin if SYNCDIVFAC = 0. For other configured SYNCDIVFAC values, the DCDCSYNCO output has 50% dutycycle.

## Power Management System (PMS)

The behavior of VGATE1N and VGATE1P during start-up and normal operation is described in section [Section 11.2.1.2.5](#).



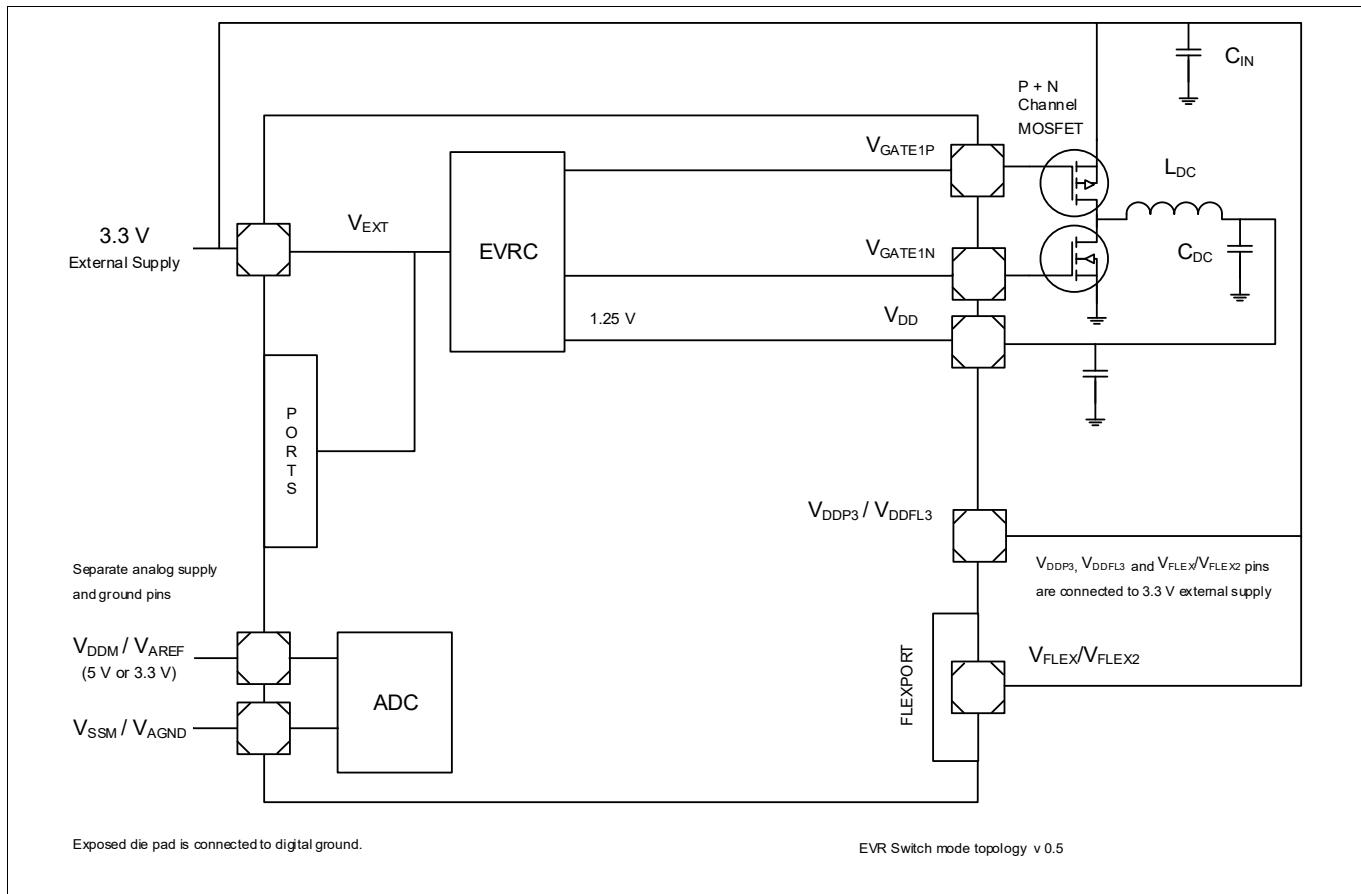
**Figure 104 EVRC Regulator Switching Frequency Spreading**

During the Start-up phase, a different control strategy is used in order to avoid current overload in the coil, overshoot of the output voltage and to charge the output capacitor in the shortest possible time. In this phase, only the P-channel MOSFET is switched ON / OFF gradually increasing the duty cycle starting at a preset value while the N-Channel MOSFET remains always OFF and behaves like a diode. This facilitates fast ramp-up to the target voltage without discharges during transients during the initial switching periods. The default switching frequency is 0.5 MHz during open loop start-up phase. The open loop operation ends when the FBADC output voltage reaches the threshold value configured in [EVRSDCTRL6.SVOTH](#) bitfield slightly below the setpoint target value. At this point the digital controller is configured and the normal closed loop operation begins and switch to 1.7 MHz switching frequency takes place. After a voltage transient (typically an overshoot), EVRC is ready and regulator output voltage is ok as indicated via [EVRSTAT.SDVOKE](#). [EVRSTAT.SDVOKE](#) remains set during RUN mode and is reset only if VDD set point was changed or VDD droop request was made during RUN mode. The start-up phase and VGATE1P and VGATE1N behavior is portrayed in [Figure 100](#). The parameters of the step down regulator is consequently updated by the Firmware after reset release to achieve a more accurate EVRC output voltage and improved performance. The step-down regulator is also later programmed with the values enumerated in [Table 290](#) so as to match the application needs and the components used. A complete parameter update is triggered explicitly by writing to [EVRSDCTRL0.UP](#) bit and it need to be ensured that all the registers are consistent before triggering the update. It need to be ensured that the EVRC registers are programmed with the right values to avoid cold PORST or overvoltage events and it cannot be always guaranteed that the EVRC recovers from wrong programming. The parameter update execution results typically in a VDD voltage transient (typically an overshoot). The parameter update is not allowed during Start-up and Low Power Mode. The droop compensation request, droop level and LPM request are taken immediately without waiting for a parameter update.

During consequent supply ramp-down phase or standby mode entry phase, the step down regulator uses the earlier programmed parameter set to ensure a graceful shut-down devoid of output voltage overshoots and coil overload. The state of both gate control outputs VGATE1P and VGATE1N during a consequent supply ramp-down phase is configured by the [EVRSDCTRL0.PGOFF](#) and [EVRSDCTRL0.NGOFF](#) register bits respectively.

## Power Management System (PMS)

The step-down regulator may be informed on anticipated load jumps so that adequate preparation can be made. The controller could lower or raise the output voltage to compensate and thus minimise voltage over-/undershoots owing to a sudden load jump. The management of voltage droop is described in Power Management [Section 11.2.3.5](#). The step down regulator issues the droop request only if the current VDD voltage level as measured by VDD FBADC is within the limits defined in [EVRSDCTRL11.DROOPVL](#) and [DROOVH](#) so as to avoid unintended resets and alarms owing to a voltage droop.



**Figure 105 EVR Switch mode topology (e) - 3.3 V single supply**

The step-down EVRC regulator also supports a Low Power Mode (LPM) which is activated on a Sleep Mode request. The activation of Low Power Mode is controlled via low power mode management and enabled via SCU\_PMTRCRS0.LPSLPEN register bit. The mode is limited to small load currents below 100 mA and has a minimum transition time of a single switching period to enter Low Power Mode. In this mode the output voltage is regulated according to a pulse frequency modulation scheme switching only the Pch. MOSFET and Nch. MOSFET acting like a diode. This mode reduces the switching losses thereby reducing the total Sleep Mode power consumption. The current mode of the step-down EVRC regulator is reflected in [EVRSTAT.EVRCMOD](#). On wake-up, the LPM mode is de-asserted and it has to be ensured that the immediate load jumps from application side are limited for the transition period as documented in the datasheet. The step-down EVRC regulator parameters are not updated during LPM mode.

During Low Power Mode, the synchronisation output (DCDCSYNCO) does not have a fixed frequency and it should be ensured that the synchronisation is not active between external regulator and TC3xx device during low power mode.

During fast ramp-down of VEXT input voltage followed by consecutive fast ramp-up bounded by datasheet parameter  $dVEXT/dT$  and during regulator start-up and standby transitions with large load jumps, voltage overshoots beyond operating conditions may occur for a short duration bounded by absolute maximum voltage

## Power Management System (PMS)

limits. During power fail of VEXT or VEVRSB input supply voltage triggering immediate LVD reset, EVRC VDD output voltage may consequently ramp down with damped oscillations leading to negative voltages on VDD rail for short duration of time (<100 us). During Standby entry , the EVRC is ramped down in a controlled manner to avoid such damped oscillations on VDD supply rail.

### 11.2.2.2.1 EVRC Frequency and Phase Synchronization to CCU6/GTM Input

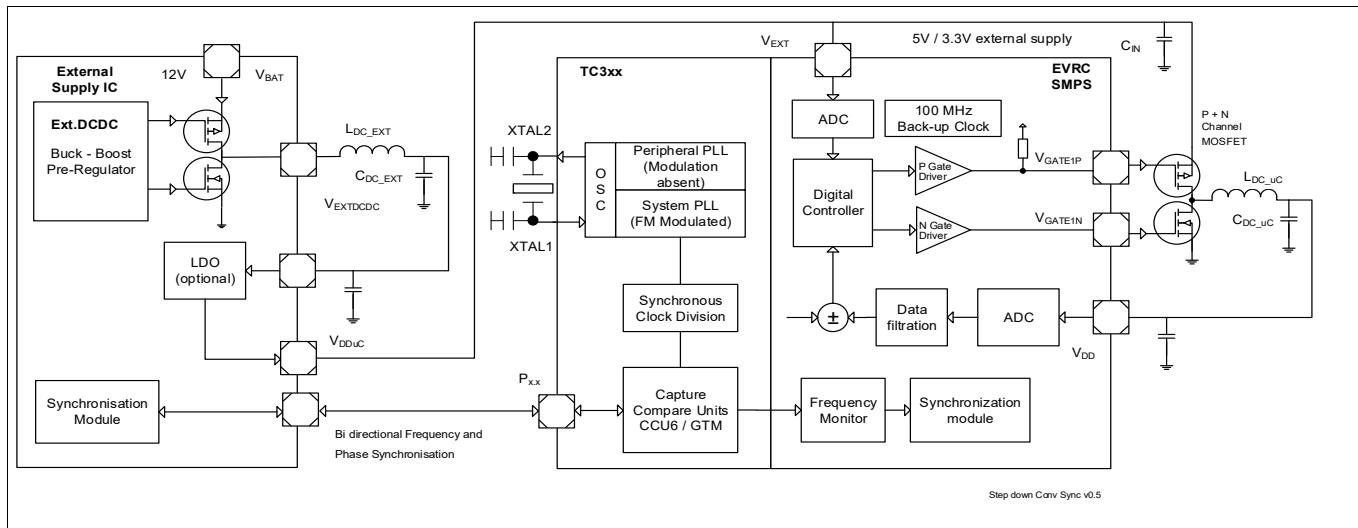
A synchronization input (DCDCSYNCI) can be provided to the EVRC SMPS regulator from CCU6 or GTM module to synchronize the frequency and the phase of the internal EVRC regulator to the external DC DC regulator. The CCU6 / GTM module provides / captures two phase synchronised PWM signals; one PWM output to the internal EVRC regulator and the other as either input from or output to an external DC DC regulator using a Port pin as shown in [Figure 106](#).

Salient aspects of the Synchronization feature are enumerated below:

- The nominal frequency of the synchronization input is 1.8 MHz. Bi-directional signal to/from external regulator is provided by the CCU6 or GTM unit.
- Frequency monitoring is carried out in the region between 1.6 MHz and 2 MHz.
- Loss of Synchronization Lock event is triggered if switching frequency range is violated. Loss of Synchronization Lock event is indicated via status bits and interrupt.
- Lock Hysteresis is provided for the Synchronization Lock signal to avoid limit cycling behavior.
- CCU60 COUT63, ATOM [0-4]\_1 and TOM [0-4]\_1 can be connected internally to LCDCDC synchronisation input.

The synchronization is supported only in the Normal PWM Mode via **EVRSCTRL11.SYNCEN** register bit and shall not be activated during Start-up or in Low Power Mode. The frequency monitor allows a maximum tolerable deviation of the synchronization input as configured in **EVRSCTRL11.SYNCMAXDEV** register bit fields. The duty cycle is not monitored, but it must be wide enough to allow proper sampling of the rising edges (minimum 2 clock cycles @100 MHz). The synchronization signal may be subjected to delays and jitter when it crosses over from the System PLL domain to the Back-up clock domain which would have an impact on the static accuracy of the EVRC regulator. The status of the synchronization lock is indicated via **EVRSTAT.SYNCLCK** bit. The loss of synchronization lock interrupt is enabled via **PMSIEN.SYNCLCK** register bit. A hysteresis is applied for the locking and unlocking, such that toggling lock behavior is avoided around the frequency monitoring limits as configured in **EVRSCTRL11.SYNCHYS** hysteresis width register bit field. Frequency spreading may be additionally activated and consequently the synchronized rising edge of the PWM signal is delayed in a random manner. Frequency step changes are limited to  $\pm 100$  kHz from CCU6 / GTM module and the dynamic frequency change in case of loss of synchronization is limited to  $\pm 200$  kHz. The component tolerances shall support the frequency range between 1.6 MHz to 2 MHz and the loss of lock dynamics. The parameters for Synchronization are documented in the datasheet.

## Power Management System (PMS)



**Figure 106 EVRC Synchronization Input**

### Synchronization Lock Procedure

- After the completion of the start-up phase or LPM mode, controller operation in EVRC PWM mode is ensured via **EVRSTAT.EVRCMOD**. The EVRC is configured for a nominal frequency of 1.8 MHz which also the reference frequency for the frequency monitor. All EVRC parameters are configured as per the intended configuration. The **EVRSDCTRL11.SYNCMAXDEV** and **SYNCHYS** bit fields are configured. Frequency spreading shall be deactivated in **EVRSDCTRL0.SDFREQSPRD** register field. Synchronization inherently provides additional jitter on switching signals which may be adequate for the EMI/EMC performance. Activating frequency spreading and synchronization jitter may increase the VDD voltage ripple.
- A synchronization input signal is provided and configured in GTM / CCU6 module and is selected via **EVRSDCTRL11.SYNCMUXSEL** bit field. GTM LCDCDCOUT signal is selected via the LCDCDCOUTSEL register in GTM module. In case of CCU60, only COUT63 is routed. A phase shifted signal may be provided to the external DCDC regulator. For High-End configurations, the register bitfield **EVRSDCOEFF0.M0SRMPCOEFF** is set to 0x09. For Low-End configurations, synchronization lock and synchronization unlock procedures shall not touch the bitfield **EVRSDCOEFF0.M0SRMPCOEFF** at all.
- When no load / line transients are ongoing, **EVRSDCTRL1.SYNCEN** bit is set to initiate the Synchronization Lock procedure.
- The frequency of the incoming synchronization signal is monitored for a single period consequently after **EVRSDCTRL1.SYNCEN** bit is set. At least one input period is required to evaluate whether the input frequency is valid. A parameter update is required to transfer the synchronization enable information.
- When the frequency of the synchronization signal is within the specified limits, immediate synchronization is started on the consecutive PWM rising edge. The synchronization is completed within 4 switching periods. The DCDC switching frequency and phase is altered and locked to the incoming signal.
- When the DCDC switching frequency is locked to the synchronization input, the **EVRSTAT.SYNCLCK** status bit is set into locked state indicating the completion of the lock procedure. The Synchronization Lock procedure takes less than 4 switching periods. After EVRC is locked to the synchronization input, the maximum delay between the rising edge of the synchronization input signal to EVRC and the consequent synchronized falling edge of the VGATE1P signal is less than 180ns.
- Depending on whether the input PWM rising edge coincides with a high or low phase of the DCDC switching period, the current switching period maybe extended or shortened.

## Power Management System (PMS)

### Synchronization Un-Lock Procedure

- Before entering LPM (Sleep) mode or Standby mode, the **EVRSDCTRL1**.SYNCEN bit shall be reset to initiate the Synchronization unlock procedure. Consequently register bitfield **EVRSDCOEFF0**.M0SRMPCOEFF, for High-End configurations, is set to 0x08. For Low-End configurations, synchronization lock and synchronization unlock procedures shall not touch the bitfield **EVRSDCOEFF0**.M0SRMPCOEFF at all.
- The **EVRSTAT**.SYNCLCK bit is set into unlocked state. The Synchronization unlock interrupt is generated if enabled. The Synchronization un-lock procedure takes less than 4 switching periods. The synchronization logic is consequently disabled.
- The synchronization input signal is consequently deactivated from GTM / CCU6 module via **EVRSDCTRL11**.SYNCMUXSEL bit field. The phase shifted signal to the external DCDC regulator from GTM / CCU6 is also deactivated.
- When the DCDC switching frequency is unlocked from the synchronization input, the **EVRSTAT**.SYNCLCK status bit is set into un-locked state indicating completion of the unlock procedure.

### Synchronization Unlock Event

- If the period of the incoming synchronization signal is respectively more than (normal period + maximum deviation) or is less than (normal period - maximum deviation); a Synchronization Unlock event is triggered.
- The **EVRSTAT**.SYNCLCK status bit is set into unlocked state. The Sync unlock interrupt is generated if enabled.
- The SMPS regulator switches to operate purely based on the internal back-up clock at default 1.8 MHz with an arbitrary phase. The Dynamic voltage deviation maybe higher during loss of synchronization (2 MHz / 1.6 MHz to 1.8 MHz jump) with consequent load jump.
- A re-lock is triggered respectively with hysteresis only if the incoming signal period is less than (normal period + maximum deviation - sync hysteresis) or more than (normal period - maximum deviation + sync hysteresis)

### 11.2.2.3 Components and Layout

The efficiency of the step-down regulator is influenced by the characteristics of the selected components and also the placement and routing of the components on the PCB. The additional external components constitute a coil, a capacitor and a complementary P-channel and N-channel MOSFET.

The coil need to have a low ESR and a relatively constant characteristic against temperature and current variations. Likewise the capacitor need to have a low ESR, a constant frequency characteristic and minimum DC voltage dependence. The capacitors may be a parallel combination of smaller and larger capacitors or capacitors of equal size for better efficiency or owing to safety considerations. The P-channel MOSFET is preferred with lower RDSON and gate capacitance values. The P-channel MOSFET gate need to be connected to VGATE1P pin. The N-channel MOSFET gate needs to be connected to VGATE1N pin for synchronous switching of the step-down regulator. In case of usage of Emulation devices, it should be taken care that the component choice also considers the current additionally drawn by Emulation RAM and additional modules.

Component characteristics would be recommended in the datasheet and is also documented in [Table 290](#).

After start-up, it need to be taken care that the register settings of EVRC are updated depending on the external components and switching frequency to ensure optimal efficiency and performance. Driver parameters are updated first followed by changing the driver slope control from hard to soft switching mode. Finally, controller parameters are updated.

It should be taken care that each supply pin in QFP packages or a pair of supply pins in BGA packages has a decoupling capacitor close to the pins. Supply pins belonging to a common supply rail shall be connected together after the respective decoupling capacitors and shall be buffered by an additional larger capacitor based on the requirements of the regulator which supplies the rail. In case of EVR33 and EVRC regulator, recommended buffer capacitors are enumerated in [Table 290](#). It should be taken care to have a low trace resistance to the

## Power Management System (PMS)

decoupling capacitors and buffer capacitors for better performance and EMI / EMC behavior. The dimensioning of the buffer capacitors is based predominantly on the load jumps triggered during reset events and stability criteria of the regulator.

**Table 289 TC3xx EVRC Regulator Component Reference**

fDCDC	TC39x ED	TC38x	TC37x	TC36x	TC35x
1.8 MHz	BSZ215C L = 3.3 uH C = 22 uF <sup>1)</sup>			BSZ215C OR BSL215C L = 3.3 uH C = 10 + 4.7 uF	BSZ215C L = 3.3 uH C = 22 uF <sup>1)</sup>
0.8 MHz	BSZ215C L = 4.7 uH C = 22+10 uF <sup>2)</sup>			-	-

1) 2 x 10 uF instead of 22 uF is also supported.

2) 3 x 10 uF instead of 22+10 uF is also supported.

**RESTRICTED**

**AURIX™ TC3xx**

**NDA Required**



---

**Power Management System (PMS)**

## Power Management System (PMS)

**Table 290 EVRC Regulator Component Reference and Register Settings**

No.	Condition	Register Update Sequence (Modes a & e)	Components (Package)
1.)	Low Current IDD < 500 mA fDCDC = 1,8 MHz VEXT < 3.63/5.5 V	<p>EVRSDCOEFF6 (Driver) = 0x004F 3802<sub>H</sub>  EVRSDCOEFF7 (Driver) = 0x0000 D02F<sub>H</sub>  EVRSDCOEFF8 (Driver) = 0x0007 3802<sub>H</sub>  EVRSDCOEFF9 (Driver) = 0000 981E<sub>H</sub>  EVRSDCCTRL7 (Driver) = 0000 00C8<sub>H</sub>  EVRSDCCTRL0 (Freq., Spreading) = 0x30360001<sub>H</sub>  EVRSDCCTRL1 (PWM mode) = 0xB69 0708<sub>H</sub>  EVRSDCCTRL0.UP = 1<sub>B</sub>  wait 20 us  check EVRSDCCTRL0.UP bit is cleared.</p> <p>EVRSDCOEFF0 (PWM mode) = 0x3608 73B6<sub>H</sub>  EVRSDCOEFF1 (PWM mode) = 0x0294 6C46<sub>H</sub>  EVRSDCCTRL0.UP = 1<sub>B</sub>  wait 20 us  check EVRSDCCTRL0.UP bit is cleared.</p> <p>EVRSDCCTRL8 (FBADC) = 0x0121 048E<sub>H</sub>  EVRSDCCTRL9 (FFADC) = 0x0000 0434<sub>H</sub>  EVRSDCCTRL10 (Short) = 0x0000 5A82<sub>H</sub>  EVRSDCCTRL11 (Droop) = 0x1207 0909<sub>H</sub>  EVRSDCCTRL4 (Start mode) = 0x0036 0009<sub>H</sub>  EVRSDCCTRL5 (Start mode) = 0xB69 0808<sub>H</sub>  EVRSDCCTRL6 (Open Loop) = 0x0023 1C94<sub>H</sub>  EVRSDCCTRL2 (LP mode) = 0x0036 033B<sub>H</sub>  EVRSDCCTRL3 (LP mode) = 0xB69 0810<sub>H</sub>  EVRSDCCTRL0 (Freq., Spreading) = 0x3036 0002<sub>H</sub>  EVRSDCOEFF0 (PWM mode) = 0x3609 74B6<sub>H</sub>  EVRSDCOEFF1 (PWM mode) = 0x0294 6C46<sub>H</sub>  EVRSDCOEFF2 (LP mode) = 0x3408 710E<sub>H</sub>  EVRSDCOEFF3 (LP mode) = 0x0294 6C44<sub>H</sub>  EVRSDCOEFF4 (Start mode) = 0x1B08 22B6<sub>H</sub>  EVRSDCOEFF5 (Start mode) = 0x0294 6C46<sub>H</sub>  EVRSDCCTRL0.UP = 1<sub>B</sub>  check EVRSDCCTRL0.UP bit is cleared.  check EVRSTAT.SDVOK is set.</p>	<p>Complementary MOSFET - BSL215C</p> <p>Inductor (3.3 uH) - CLF5030NIT- 3R3N-D or LTF3020T-3R3N-H/D (long term availability not assured) or RLF7030T-3R3M4R1-T or TFM252012ALMA3R3MTAA (under analysis)</p> <p>Output Capacitor (10 uF + 4,7uF) - CGA6M3X7R1C106K + CGA5L3X7R1C475K</p> <p>Input Capacitor (6.8 uF) - CGA5L1X7R1C685M</p>

## Power Management System (PMS)

**Table 290 EVRC Regulator Component Reference and Register Settings (cont'd)**

No.	Condition	Register Update Sequence (Modes a & e)	Components (Package)
2.)	IDD < 1,5A fDCDC = 1,8 MHz VEXT < 3.63/5.5 V	<p>EVRSDCOEFF6 (Driver) = 0x0087 3802<sub>H</sub>  EVRSDCOEFF7 (Driver) = 0x0000 D066<sub>H</sub>  EVRSDCOEFF8 (Driver) = 0x0007 3802<sub>H</sub>  EVRSDCOEFF9 (Driver) = 0x0000 9826<sub>H</sub>  EVRSDCCTRL7 (Driver) = 0x0000 00C9<sub>H</sub>  EVRSDCCTRL0 (Freq., Spreading) = 0x30360001<sub>H</sub>  EVRSDCCTRL1 (PWM mode) = 0xB69 0708<sub>H</sub>  EVRSDCCTRL8 (FBADC) = 0x0121048E<sub>H</sub>  EVRSDCCTRL0.UP = 1<sub>B</sub>  wait 20 us  check EVRSDCCTRL0.UP bit is cleared.</p> <p>EVRSDCOEFF0 (PWM mode) = 0x3508 73B6<sub>H</sub>  EVRSDCOEFF1 (PWM mode) = 0x0294 6C46<sub>H</sub>  EVRSDCCTRL0.UP = 1<sub>B</sub>  wait 20 us  check EVRSDCCTRL0.UP bit is cleared.</p> <p>EVRSDCCTRL0 (Freq., Spreading) = 0x3036 0002<sub>H</sub>  EVRSDCCTRL2 (LP mode) = 0x0036 033B<sub>H</sub>  EVRSDCCTRL3 (LP mode) = 0xB69 0810<sub>H</sub>  EVRSDCOEFF2 (LP mode) = 0x3408 710E<sub>H</sub>  EVRSDCOEFF3 (LP mode) = 0x0294 6C44<sub>H</sub>  EVRSDCCTRL4 (Start mode) = 0x0036 0009<sub>H</sub>  EVRSDCCTRL5 (Start mode) = 0xB69 0808<sub>H</sub>  EVRSDCCTRL6 (Open Loop) = 0x0023 1C94<sub>H</sub>  EVRSDCOEFF4 (Start mode) = 0x1B08 22B6<sub>H</sub>  EVRSDCOEFF5 (Start mode) = 0x0294 6C46<sub>H</sub>  EVRSDCCTRL9 (FFADC) = 0x0000 0434<sub>H</sub>  EVRSDCCTRL10 (Short) = 0x0000 5A82<sub>H</sub>  EVRSDCCTRL11 (Droop) = 0x1207 0909<sub>H</sub>  EVRSDCCTRL0.UP = 1<sub>B</sub>  check EVRSDCCTRL0.UP bit is cleared.  check EVRSTAT.SDVOK is set.</p>	<p>Complementary MOSFET - BSZ15DC02KD / BSZ215C</p> <p>Inductor (3.3 uH) - CLF6045NIT-3R3N-D or LTF5022T-3R3N2R5-H/D (long term availability not assured) or TFM252012ALMA3R3MTAA (under analysis)</p> <p>Output Capacitor (22 uF or 2x10 uF) - CGA6P1X7R1C226M or 2 x CGA6M3X7R1C106K</p> <p>Input Capacitor (10 uF) - CGA6M3X7R1C106K</p>

## Power Management System (PMS)

**Table 290 EVRC Regulator Component Reference and Register Settings (cont'd)**

No.	Condition	Register Update Sequence (Modes a & e)	Components (Package)
3.)	IDD < 1,5 A fDCDC = 0.8 MHz VEXT < 3.63/5.5 V	EVRSDCOEFF6 (Driver) = 0x0087 3802 <sub>H</sub> EVRSDCOEFF7 (Driver) = 0x0000 D066 <sub>H</sub> EVRSDCOEFF8 (Driver) = 0x0007 3802 <sub>H</sub> EVRSDCOEFF9 (Driver) = 0x0000 9826 <sub>H</sub> EVRSDCTRL7 (Driver) = 0x0000 00C9 <sub>H</sub> EVRSDCTRL0 (Freq., Spreading) = 0x307C 0001 <sub>H</sub> EVRSDCTRL1 (PWM mode) = 0x0B69 0708 <sub>H</sub> EVRSDCTRL8 (FBADC) = 0x0121 048E <sub>H</sub> EVRSDCTRL0.UP = 1 <sub>B</sub> wait 20 us check EVRSDCTRL0.UP bit is cleared.  EVRSDCOEFF0 (PWM mode) = 0x3408 7336 <sub>H</sub> EVRSDCTRL0.UP = 1 <sub>B</sub> wait 20 us check EVRSDCTRL0.UP bit is cleared.  EVRSDCTRL0 (Freq., Spreading) = 0x307C 0002 <sub>H</sub> EVRSDCOEFF0 (PWM mode) = 0x3408 7236 <sub>H</sub> EVRSDCOEFF1 (PWM mode) = 0x0294 6C46 <sub>H</sub> EVRSDCTRL2 (LP mode) = 0x0036 033B <sub>H</sub> EVRSDCTRL3 (LP mode) = 0x0B69 0810 <sub>H</sub> EVRSDCOEFF2 (LP mode) = 0x3408 710E <sub>H</sub> EVRSDCOEFF3 (LP mode) = 0x0294 6C44 <sub>H</sub> EVRSDCTRL4 (Start mode) = 0x0036 0009 <sub>H</sub> EVRSDCTRL5 (Start mode) = 0x0B69 0808 <sub>H</sub> EVRSDCTRL6 (Open Loop) = 0x0023 1C94 <sub>H</sub> EVRSDCOEFF4 (Start mode) = 0x1B08 22B6 <sub>H</sub> EVRSDCOEFF5 (Start mode) = 0x0294 6C46 <sub>H</sub> EVRSDCTRL9 (FFADC) = 0x0000 0434 <sub>H</sub> EVRSDCTRL10 (Short) = 0000 5A82 <sub>H</sub> EVRSDCTRL11 (Droop) = 0x1207 0909 <sub>H</sub> EVRSDCTRL0.UP = 1 <sub>B</sub> check EVRSDCTRL0.UP bit is cleared. check EVRSTAT.SDVOK is set.	Complementary MOSFET - BSZ15DC02KD / BSZ215C  Inductor (4.7 uH) - CLF6045NIT-4R7N-D or LTF5022T-4R7N2R0-H/D (long term availability not assured)  Output Capacitor (22 uF + 10uF) or (3x10 uF) - CGA6P1X7R1C226M + CGA6M3X7R1C106K or 3 x CGA6M3X7R1C106K  Input Capacitor (10 uF) - CGA6M3X7R1C106K

## Power Management System (PMS)

**Table 290 EVRC Regulator Component Reference and Register Settings (cont'd)**

No.	Condition	Register Update Sequence (Modes a & e)	Components (Package)
4.)	Low Current IDD < 700 mA fDCDC = 0.8 MHz VEXT < 3.63/5.5 V	<p>EVRSDCOEFF6 (Driver) = 0x004F 3802<sub>H</sub>  EVRSDCOEFF7 (Driver) = 0x0000 D02F<sub>H</sub>  EVRSDCOEFF8 (Driver) = 0x0007 3802<sub>H</sub>  EVRSDCOEFF9 (Driver) = 0x0000 981E<sub>H</sub>  EVRSDCCTRL7 (Driver) = 0x0000 00C8<sub>H</sub>  EVRSDCCTRL0 (Freq., Spreading) = 0x307C 0001<sub>H</sub>  EVRSDCCTRL1 (PWM mode) = 0xB69 0708<sub>H</sub>  EVRSDCCTRL8 (FBADC) = 0x0121 048E<sub>H</sub>  EVRSDCCTRL0.UP = 1<sub>B</sub>  wait 20 us  check EVRSDCCTRL0.UP bit is cleared.</p> <p>EVRSDCOEFF0 (PWM mode) = 0x3508 73B6<sub>H</sub>  EVRSDCOEFF1 (PWM mode) = 0x2294 6C46<sub>H</sub>  EVRSDCCTRL0.UP = 1<sub>B</sub>  wait 20 us  check EVRSDCCTRL0.UP bit is cleared.</p> <p>EVRSDCCTRL9 (FFADC) = 0x0000 0434<sub>H</sub>  EVRSDCCTRL10 (Short) = 0000 5A82<sub>H</sub>  EVRSDCCTRL11 (Droop) = 0x141F 0909<sub>H</sub>  EVRSDCCTRL4 (Start mode) = 0x0036 0009<sub>H</sub>  EVRSDCCTRL5 (Start mode) = 0xB69 0808<sub>H</sub>  EVRSDCCTRL6 (Open Loop) = 0x0023 1C94<sub>H</sub>  EVRSDCCTRL2 (LP mode) = 0x0036 033B<sub>H</sub>  EVRSDCCTRL3 (LP mode) = 0xB69 0810<sub>H</sub>  EVRSDCCTRL1 (PWM mode) = 0xB69 0708<sub>H</sub>  EVRSDCCTRL0 (Freq., Spreading) = 0x307C 0002<sub>H</sub>  EVRSDCOEFF0 (PWM mode) = 0x3508 7236<sub>H</sub>  EVRSDCOEFF1 (PWM mode) = 0xA94 6C46<sub>H</sub>  EVRSDCOEFF2 (LP mode) = 0x3408 710E<sub>H</sub>  EVRSDCOEFF3 (LP mode) = 0x294 6C44<sub>H</sub>  EVRSDCOEFF4 (Start mode) = 0x1B08 22B6<sub>H</sub>  EVRSDCOEFF5 (Start mode) = 0x294 6C46<sub>H</sub>  EVRSDCCTRL0.UP = 1<sub>B</sub>  check EVRSDCCTRL0.UP bit is cleared.  check EVRSTAT.SDVOK is set.</p>	<p>Complementary MOSFET - BSZ15DC02KD / BSZ215C</p> <p>Inductor (4.7 uH) - CLF6045NIT-4R7N-D or LTF5022T-4R7N2R0-H/D (long term availability not assured)</p> <p>Output Capacitor (20uF) - 2 x CGA6M3X7R1C106K</p> <p>Input Capacitor (6.8 uF) - CGA5L1X7R1C685M</p>

**Table 291 EVR33 External Component Reference**

No.	Condition	Optimal Register Values	Components (Package)
1.)	IDDP3 < 100 mA		Output Buffer capacitor (2,2 uF)

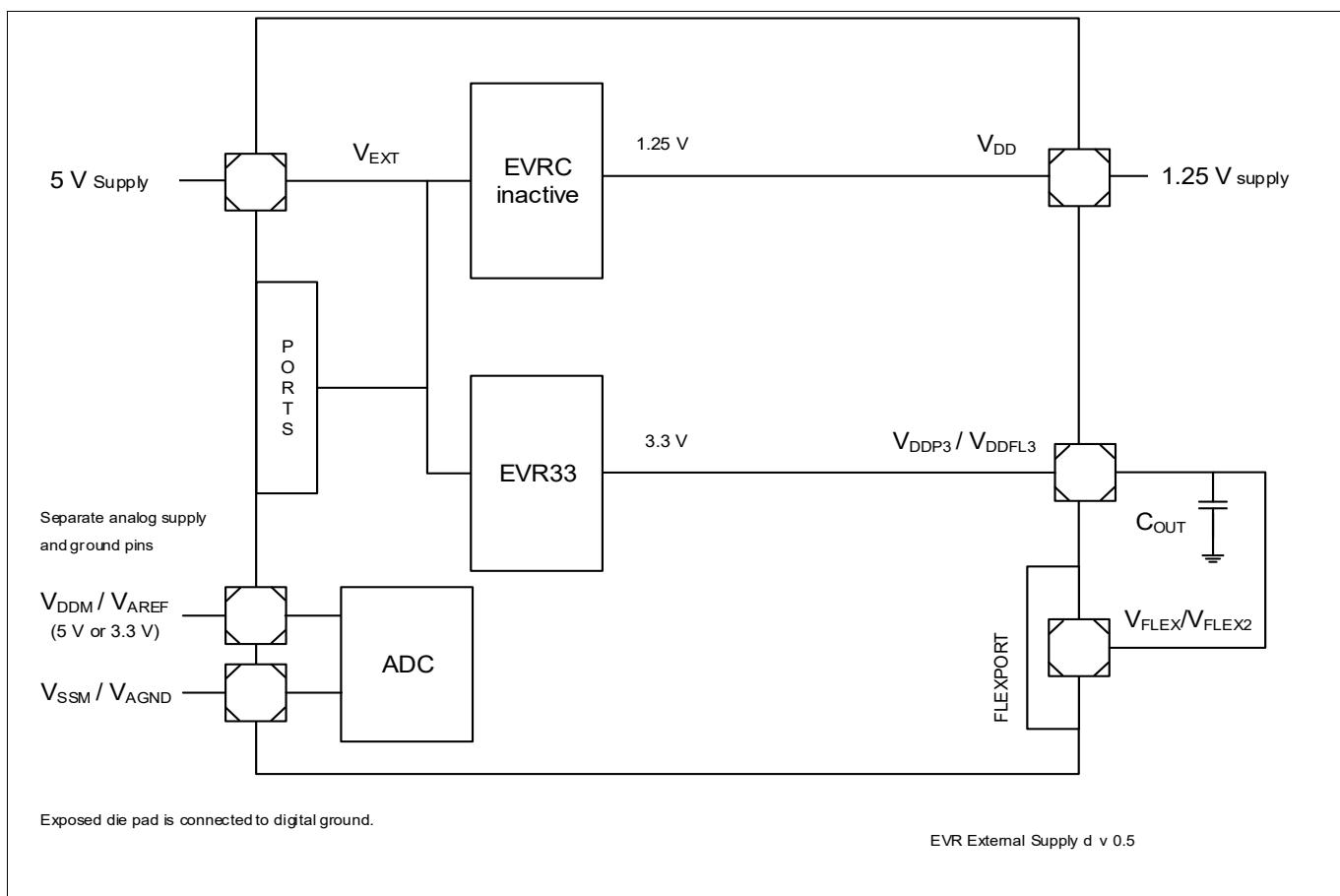
## Power Management System (PMS)

### 11.2.2.4 External Supply Modes

The external supply modes involve deactivating any or both of the EVRC and EVR33 regulators. In this mode, EVR33 is disabled via the HWCFG[1] configuration pin and the EVRC is disabled via the HWCFG[2] configuration pin respectively.

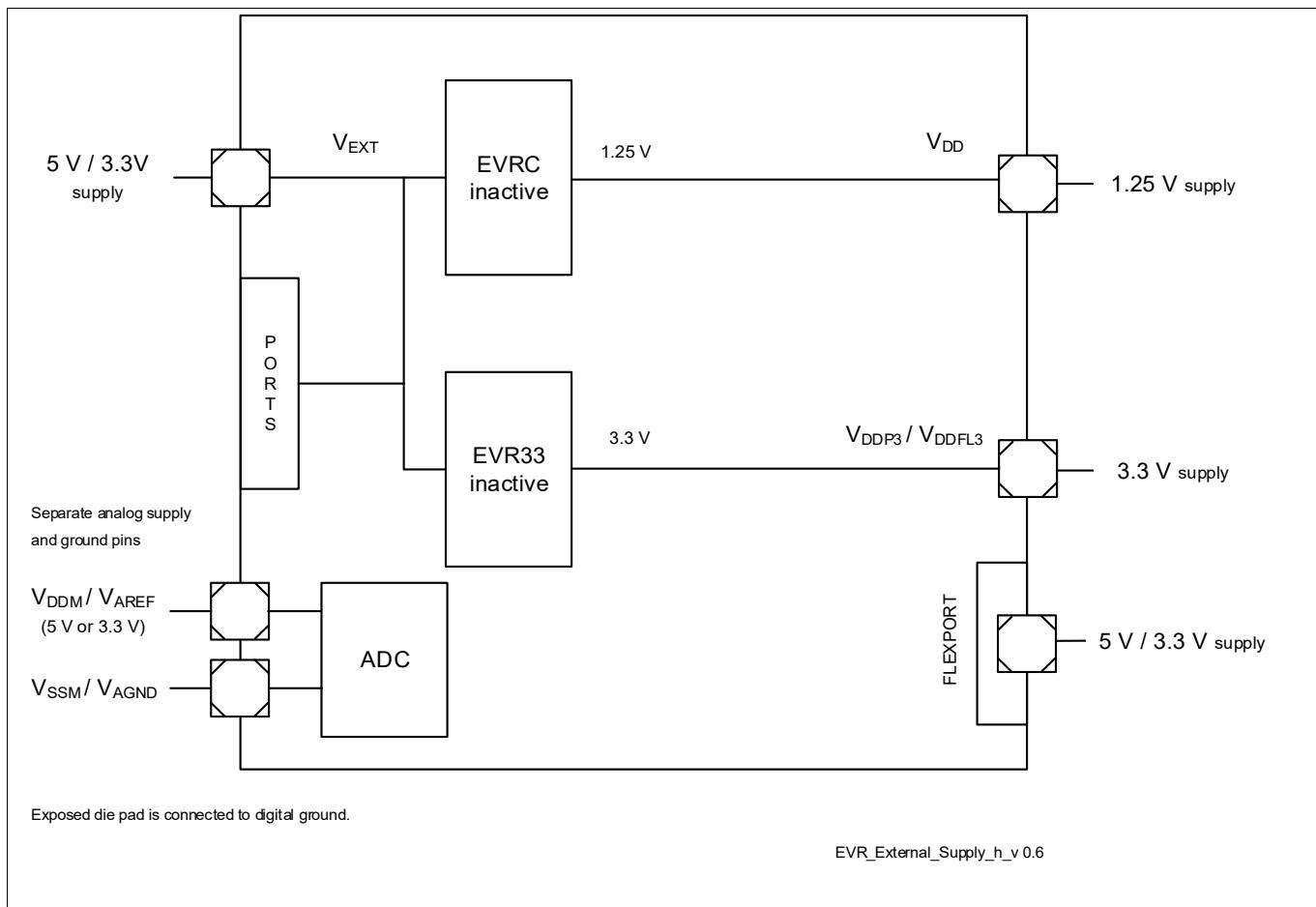
Following external supply modes are supported.

- VEXT = 5 V and VDD supplied externally. VDDP3 is generated using the EVR33 regulator as shown in [Figure 107](#).
- VEXT = 5 V or 3.3 V and VDDP3 is supplied externally. VDD is generated using the EVRC regulator.
- VEXT, VDDP3 and VDD are all supplied externally as shown in [Figure 108](#).



**Figure 107 External Supply mode (d) - VEXT and VDD externally supplied**

## Power Management System (PMS)



**Figure 108 External Supply mode (h) -  $V_{EXT}$ ,  $V_{DDP3}$  and  $V_{DD}$  externally supplied**

## Power Management System (PMS)

### 11.2.2.5 Supply Voltage Monitoring

The PMS module implements a staggered voltage monitoring build upon a primary and a secondary monitor providing adequate redundancy to meet safety requirements. The primary monitor ensures that the micro controller is put into a cold PORST reset state when the lowest operational voltage thresholds are violated. The secondary monitor serves as an additional safety monitor providing over- and under-voltage alarms for multiple supply rails. Monitors are realized using dedicated 8 bit ADC converters and result comparators.

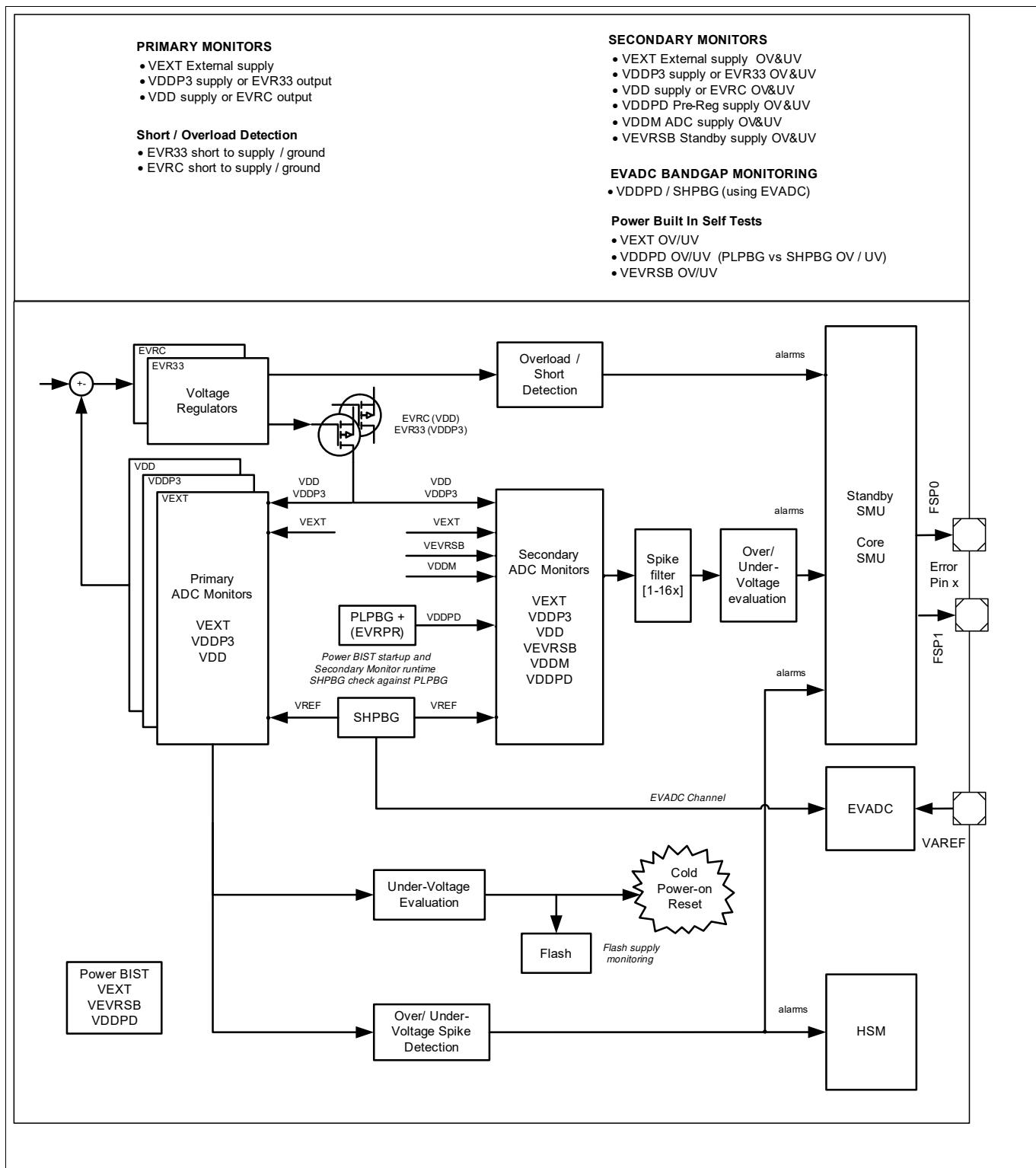


Figure 109 Supply Monitor Overview

## Power Management System (PMS)

### 11.2.2.5.1 Primary under-voltage monitors and Cold PORST

Primary under-voltage monitoring of the external VEXT supply, VDDP3 / EVR33 supply and VDD / EVRC supply are inherently carried out to ensure proper functioning of the system. The thresholds for the primary monitors represent the lowest possible thresholds for the correct functioning of the system. The threshold and tolerance is documented in datasheet as VxxPRIUV parameter. In case of violation of these thresholds, cold PORST is activated and PORST pin is pulled low (strong current sink) thus setting the device into reset state. Following the reset release and firmware boot, it can be inferred from STBYR, EVRC, EVR33 or SWD bits in RSTSTAT register as to whether the violation of these thresholds led to the previous reset. The primary under-voltage monitoring is kept active even if the respective EVRs have been disabled and the supply is provided externally as shown in [Table 292](#). The thresholds are trimmed and the monitoring is activated or deactivated via the [EVRRSTCON](#) register. The user shall not modify the default values of the [EVRRSTCON](#) register, as any alteration of the primary reset monitoring violates the operational conditions of the microcontroller and may lead to unexpected behavior during the dynamic undershoot regulation or during the power down sequence.

The cold PORST is asserted when the supply voltage drops below [EVRRSTCON.RSTxTRIM](#) value. During cold PORST reset release, to avoid consecutive toggling PORST during slow supply ramp-ups, a voltage hysteresis is supported. The cold PORST is de-asserted or released when the supply rises above ([EVRRSTCON.RSTxTRIM](#) + Hysteresis) value. The PORST pin is driven low for a minimum nominal time of 10 us on recognition of cold PORST irrespective whether the voltages have been immediately restored so that there is adequate time to recognise it externally.

Further more, additional power-on detectors are available for VEVRSB supply (supplied by VEXT), VEXT supply (supplied by VEVRSB) and VDDPD internal supply (via VDDPD POR monitor) to ensure a proper minimum-power detection, robust start-up and standby operation. Undervoltage of VDDPD internal supply and external VEXT supply will lead to the assertion of the LVD (Low Voltage Detector) reset. Undervoltage of external VEVRSB supply will lead to the assertion of the LVD reset indirectly via the VDDPD POR monitor as VDDPD is generated from VEVRSB. Assertion of LVD reset is reflected in RSTSTAT.STBYR bit and can be evaluated in the next start-up. After a normal supply start-up, only STBYR and PORST bits in RSTSTAT register are set.

The primary Supply WatchDog (SWD monitor) monitors the ramp-up of external VEXT supply voltage and keeps the micro controller in cold Power On Reset state as long as the supply has not reached the operational region. Likewise, it also allows detecting ramp-down or brown out conditions of external supply so that the device can be brought into a cold Power On Reset state when the voltage has dropped below the lowest operational threshold. Nevertheless, It is recommended to monitor externally all supplies generated external to the micro controller and to assert PORST reset pin in case of violation of the lowest operational limits. The pass device dropout voltage should be taken into consideration when setting these limits. In case of 5 V nominal external supply and 3.3 V in turn being generated by the internal EVR33 LDO regulator, the external supply shall maximum drop during normal RUN mode considering adequate pass device dropout as documented in datasheet.

The external VEXT supply, VDD / EVRC and VDDP3 / EVR33 supplies are measured by Primary Monitor ADCs and the measured value is updated in [EVRADCSTAT](#) register after conversion completion at every PMS clock cycle.

In case of primary monitor violation, respective status bits are set to indicate the event as shown in [Table 292](#). These bits maybe evaluated during consequent start-up after cold power-fail reset to recognize which among the supply rails had the power-fail.

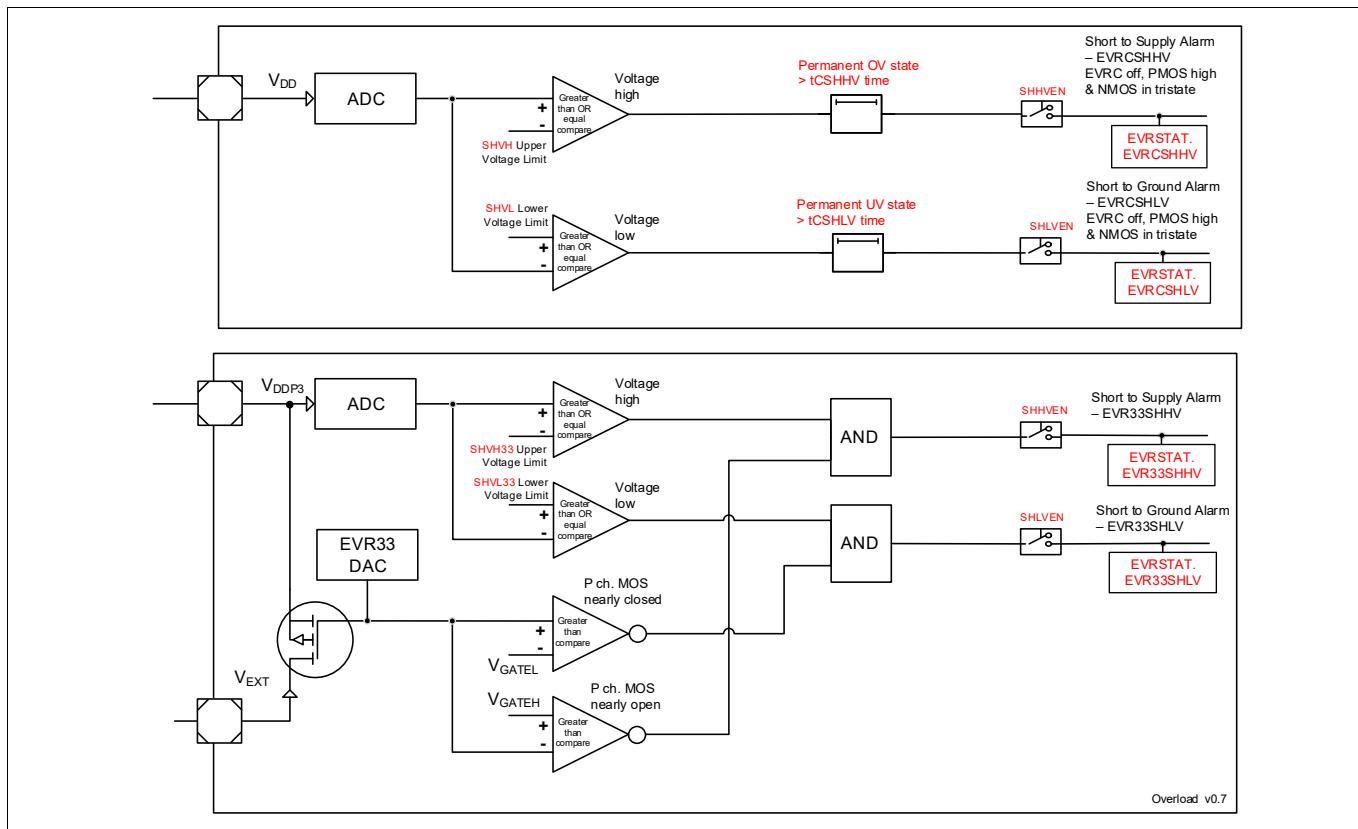
The violation of the primary under-voltage and over-voltage operational limits of VEXT, VDDP3 and VDD supply rails is communicated to the HSM module and to the SMU. HSM module may lock access to EVR registers via SLCK bit so that supply generation cannot be influenced by other masters. This is to ensure that trojan programs do not manipulate the supplies to gain access to the system. VEXT, VDDP3 and VDD rail primary monitor measurements are compared with [HSMOVMON](#) and [HSMUVMON](#) thresholds and alarms are routed to the HSM module and to the SMU (as shown in [Figure 109](#)). The violations are indicated in [EVRADCSTAT](#) status flags. The unfiltered primary monitor ADC measurements are used to detect power spikes on the main supply rails and consequently alarms are provided to HSM and SMU. Each primary monitor ADC tracking speed is bounded by the

## Power Management System (PMS)

maximum supply slope of a single LSB step every nominal 25 MHz ADC clock cycle. This results in a maximum tracking speed of 500V/ms (20mV LSB/40ns) for VEXT SWD primary monitor, 375V/ms (15mV LSB/40ns) for VDDP3 primary monitor

A voltage based short detection scheme is enabled for EVRC via **EVRSDCTRL10.SHLVEN / SHHVEN** register bit fields. The short detection scheme for EVRC output is as portrayed in [Figure 110](#). VDD FBADC result is compared against SHVL and SHHV thresholds. If the low voltage or high voltage condition occurs continuously for more than tCSHLV or tCSHHV duration, the respective voltage alarms are activated and are indicated by **EVRSTAT.EVRxSHHV** and **EVRSTAT.EVRxSHLV** register status bits. If the low voltage or high voltage condition disappears before tCSHLV or tCSHHV expiry, then tCSHLV or tCSHHV timers are reset. The recovery from EVRC short switch-off state is possible only with a renewed ramp-up of VEVRSB and VEXT supply rails. EVRC Short signal is filtered for 6 consecutive values using a spike filter and the filtered signal leads to EVRC switch off.

A short detection scheme may be activated for EVR33 via .SHLVEN / SHHVEN bits. The short detection scheme for EVR33 is portrayed in [Figure 110](#). Short to higher voltage is deduced when the voltage regulator control output or pass device gate voltage has saturated at the lower limit and at the same time the regulator voltage output has crossed the absolute maximum limit. Short to lower voltage is deduced when the voltage regulator control output or pass device gate voltage has saturated at the upper limit and at the same time the regulator voltage output has stayed at the minimum limit. In both cases the respective alarms are activated and indicated by **EVRSTAT.EVR33SHHV** and **EVRSTAT.EVR33SHLV** register bits.



**Figure 110 Short to Supply and Ground Detection**

## Power Management System (PMS)

### 11.2.2.5.2 Secondary over- and under-voltage monitors and alarm generation

Additional secondary over-voltage and under-voltage monitoring against programmable thresholds is provided for all supplied and generated voltages. The secondary monitors are based on a secondary bandgap reference independent from the primary band-gap reference. The monitored voltages include the external VEXT supply voltage, VDDP3 / EVR33 supply, VDD / EVRC supply, external VEVRSB supply voltage, external VDDM ADC supply voltage and the internally generated VDDPD Pre- Regulator output voltage as shown in [Figure 112](#) and [Figure 113](#). The secondary voltage monitors are kept active even if the respective EVRs have been disabled and the supply is provided externally as shown in [Table 292](#). In case of a threshold violation, an SMU alarm event is generated. The threshold and tolerance is documented in datasheet as VxxMON parameter.

The secondary monitor violation is notified depending on the direction of voltage transition as programmed in [EVROMONCTRL](#) register. The appropriate thresholds for voltage monitoring can be programmed in the [EVROVMON](#), [EVROVMON2](#), [EVRUVMON](#) and [EVRUVMON2](#) registers. These can be calculated by linear interpolation based on multiple voltage levels and corresponding thresholds provided in VxxMON datasheet parameters. In case of an active monitoring violation, respective status flags are set in the [EVRSTAT](#) register. It can be inferred from OVC, OV33, OVSWD, OVPRE, OVS and OVDDM bits in [EVRSTAT](#) register as to whether over-voltage thresholds for the respective voltage domains were violated. Likewise, it can be inferred from UVC, UV33, UVSWD, UVPRE, UVS and UVDDM bits in [EVRSTAT](#) register as to whether under-voltage thresholds were violated. The respective status bits may be evaluated to differentiate between an over-voltage or an under-voltage event and to recognize which supply rail had triggered the alarm event to SMU as shown in [Table 292](#). The secondary monitor measurement latency to measure all 6 supply rails is documented in datasheet as tMON parameter. The supply rails are converted one after another in a continuous scan mode. It is also possible to deactivate individually the secondary monitors in [EVROMONCTRL](#) register. If the respective OVMOD and UVMOD bits are set to 00, then the ADC conversion for the particular supply rail is skipped by the Secondary Monitor and (tMON/6) time is respectively reduced from the total conversion time.

The [EVRUVMON2.VDDMLVLSEL](#) bit-field shall not be modified by the application software (as it is not related to the secondary monitoring thresholds). The application SW shall always read out the default value of [EVRUVMON2.VDDMLVLSEL](#) and write it back unmodified together with any new undervoltage monitoring threshold information in the [EVRUVMON2](#) register.

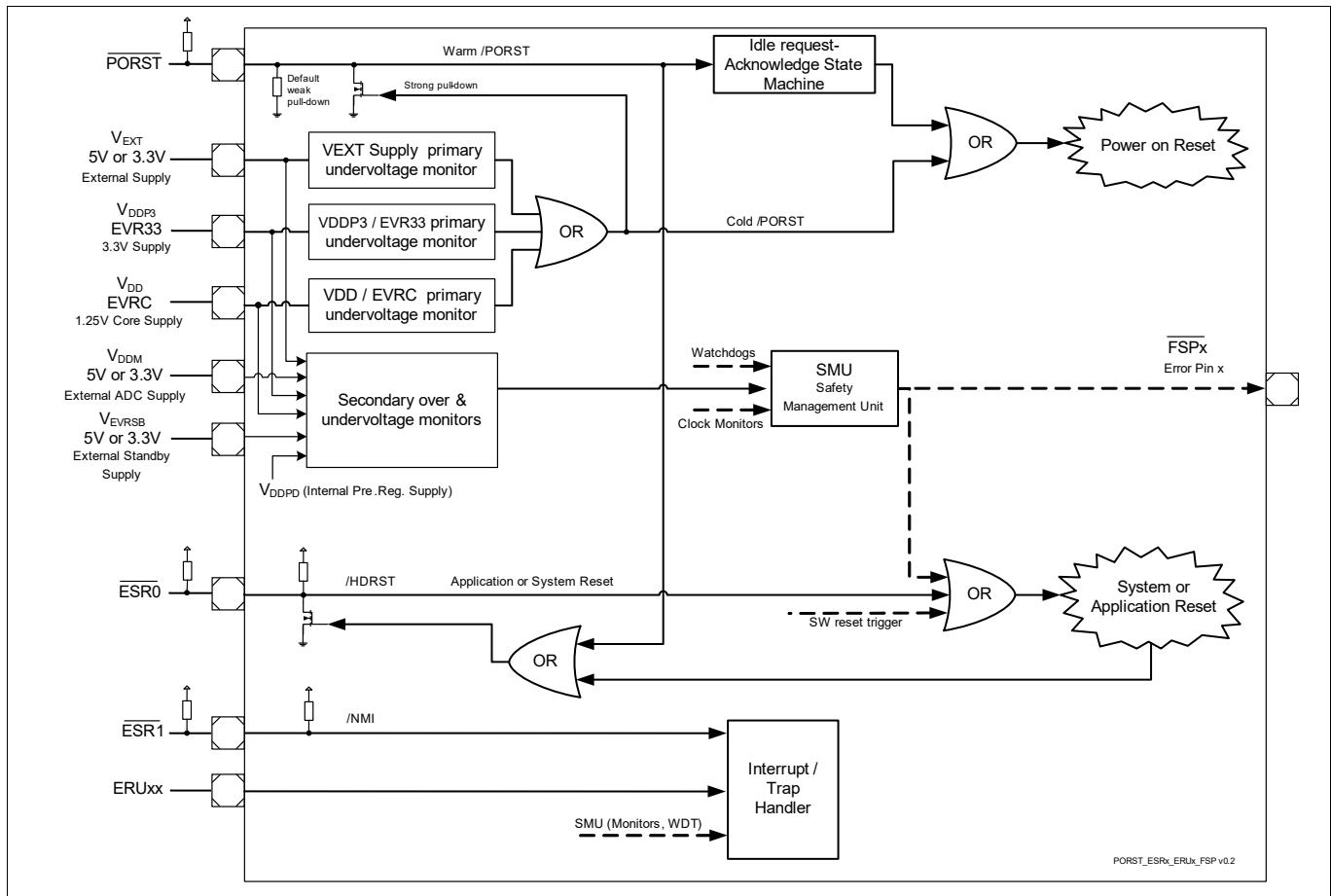
The monitored voltages, namely the VEXT, VDDP3, VDD, VDDPD, VEVRSB, and VDDM supplies are measured by Secondary Monitor ADCs and the actual measured value is updated in [EVROMONSTAT1](#) and [EVROMONSTAT2](#) register after conversion completion at regular intervals. Spike filtering of consecutive ADC results are used to generate alarm to SMU and also used for the filtered values indicated in [EVROMONSTAT1 / EVROMONSTAT2](#) registers as configured via adjustable filter coefficients in [EVROMONFILT.xx FIL](#) bit fields. In case VDDM supply voltage drops below 500 mV outside the operational limits, then Secondary monitor stops converting and the activity counter [EVROMONSTAT1.ACTVCNT](#) freezes at the last value.

In case of over-voltage supply alarms, it may be ensured that the supply to the device is switched off to avoid damage. The Error Pin Fail Safe Protocol ensures that the over-voltage condition is communicated to the external regulator even when TC3xx is in warm reset state.

After start-up, it may happen that supply over- or under-voltage alarms may already have been triggered depending on residual start-up voltages or supply dynamics. Likewise during [EVROMONCTRL](#) or [EVROMONFILT](#) reconfigurations, spurious alarms may be raised depending on filter state and changed configuration. Therefore before activating SMU alarm generation or triggering latent fault supply alarm tests, the secondary monitors and filters need to be completely reset. It needs to be ensured that SMU alarms and associated interrupts are foremost deactivated in [EVROMONCTRL / EVROMONFILT / PMSIEN](#) registers, then filters are cleared via [EVROMONFILT.CLRFIL = 1](#), alarms and interrupts are then consequently re-configured to the intended voltage level and filter settings in [EVROMONCTRL / EVROMONFILT](#) registers followed by activation of filters via [EVROMONFILT.CLRFIL = 0](#). A delay time of 4 us has to be awaited before alarm activation after configuration is changed in [EVROMONCTRL / EVROMONFILT](#) registers.

## Power Management System (PMS)

In case of application and system resets, PMS alarms happening during the respective reset shutdown and release will be reflected in SMU\_stdby AGX alarm status registers and consequently SMU\_stdby FSP reaction may be triggered if so configured in SMU\_stdby AGFSP.FEx registers. On the contrary, PMS alarms occurring during warm reset phase will be not be latched in SMU\_core AGX alarm status registers as they are in reset state. Furthermore, alarms which have occurred during the reset phase would not be consequently forwarded to the SMU\_core on reset release.



**Figure 111 Monitoring and Reset Pins**

**Power Management System (PMS)**
**Table 292 Voltage Monitoring**

<b>Supply Pin / Rail</b>	<b>Primary Under-voltage Monitor State (ON/OFF) Status Registers set on Under-voltage</b>	<b>Secondary Over &amp; Under-voltage Monitor State (ON/OFF) Status Registers</b>	<b>Supply Range V</b>	<b>Is the Pin supplied</b>
RUN or SLEEP system mode during supply modes a,d,e & h.				
$V_{EXT}$	<p>ON. RSTSTAT.SWD set if <math>V_{EXT}</math> drops below VEXTPRIUV limit triggering cold PORST. During cold start-up on an initial <math>V_{EXT}</math> ramp-up, RSTSTAT.SWD is not set. RSTSTAT.PORST bit implicitly set as cold PORST would trigger also warm PORST.</p> <p>RSTSTAT.STBYR set if <math>V_{EXT}</math> drops below VLVRST5 voltage limit.</p> <p>EVRSTAT.RSTSVD shows current status. EVRAADCSTAT.ADCSVD shows ADC result. EVRAADCSTAT.OVSV (HSM &amp; SMU alarm) EVRAADCSTAT.UVSV (HSM &amp; SMU alarm)</p>	<p>ON EVRSTAT.OVSWD (SMU alarm) EVRSTAT.UVSWD (SMU alarm) EVROMONSTAT1.ADCSVD</p>	2.97-5.50 V	External 5V or 3.3V Supply to be provided
$V_{DDP3}$	<p>ON RSTSTAT.EVR33 set if <math>V_{DDP3}</math> drops below VDDP3PRIUV limit triggering cold PORST. During cold start-up on an initial <math>V_{DDP3}</math> ramp-up, RSTSTAT.EVR33 is not set. RSTSTAT.PORST bit implicitly set as cold PORST would trigger also warm PORST.</p> <p>EVRSTAT.RST33 shows current status. EVRAADCSTAT.ADC33V shows ADC result. EVRAADCSTAT.OV33 (HSM &amp; SMU alarm) EVRAADCSTAT.UV33 (HSM &amp; SMU alarm)</p>	<p>ON EVRSTAT.OV33 (SMU alarm) EVRSTAT.UV33 (SMU alarm) EVROMONSTAT1.ADC33V</p>	2.97-3.63 V	EVR33 active or external 3.3V supply to be provided
$V_{DD}$	<p>ON RSTSTAT.EVRC set if <math>V_{DD}</math> drops below VDDPRIUV limit triggering cold PORST. During cold start-up on an initial <math>V_{DD}</math> ramp-up, RSTSTAT.EVRC is not set. RSTSTAT.PORST bit implicitly set as cold PORST would trigger also warm PORST. EVRSTAT.RSTC shows current status.</p> <p>EVRAADCSTAT.ADCCV shows ADC result. EVRAADCSTAT.OVC (HSM &amp; SMU alarm) EVRAADCSTAT.UVC (HSM &amp; SMU alarm) (RSTSTAT.PORST bit implicitly set)</p>	<p>ON EVRSTAT.OVC (SMU alarm) EVRSTAT.UVC (SMU alarm) EVROMONSTAT1.ADCCV</p>	1.125-1.375 V	EVRC active or external 1.25V supply to be provided

## Power Management System (PMS)

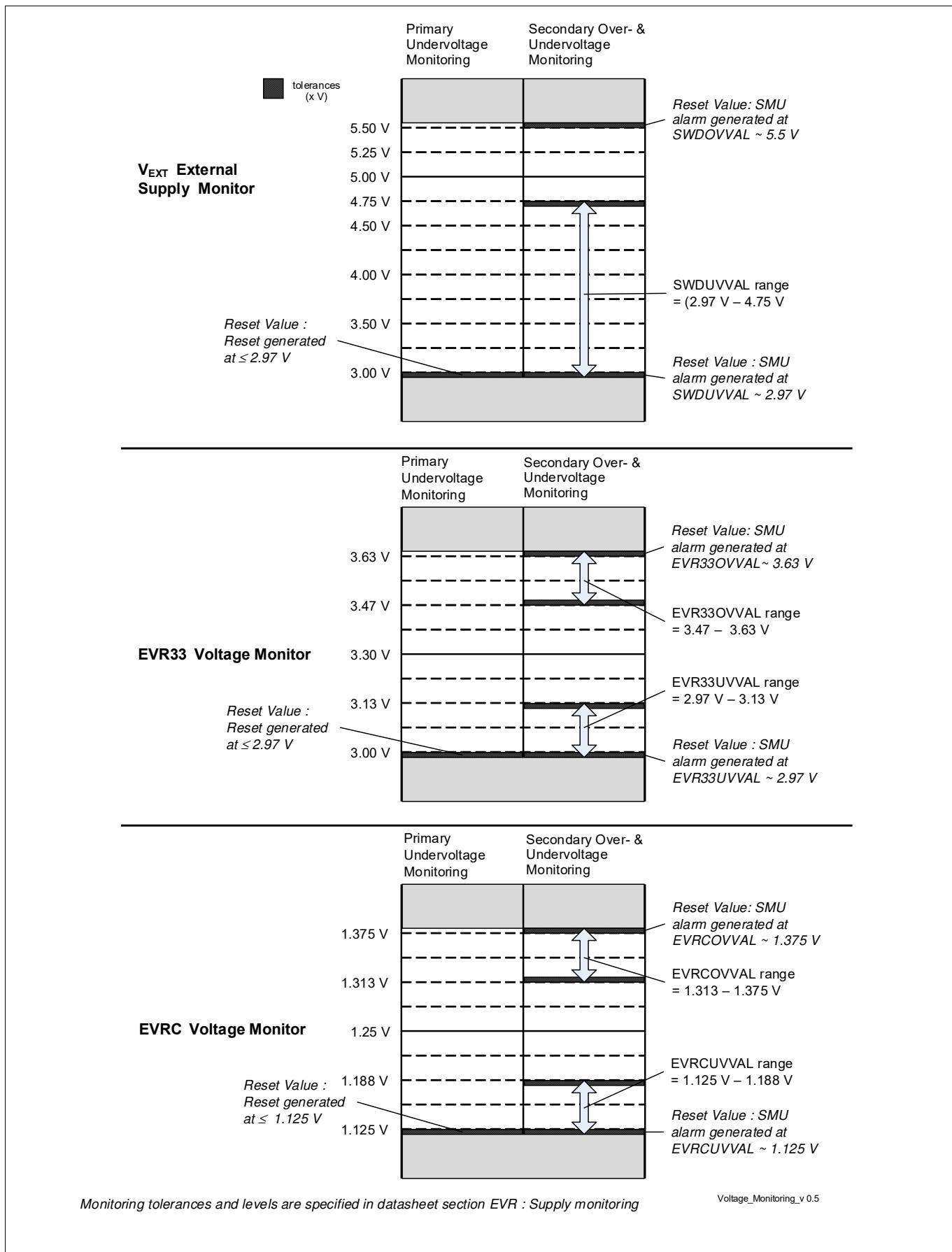
**Table 292 Voltage Monitoring (cont'd)**

Supply Pin / Rail	Primary Under-voltage Monitor State (ON/OFF) Status Registers set on Under-voltage	Secondary Over & Under-voltage Monitor State (ON/OFF) Status Registers	Supply Range V	Is the Pin supplied
$V_{EVRSB}$	ON.(via VEVRSB detector) RSTSTAT.STBYR set if $V_{EVRSB}$ drops below VLDRSTSB voltage limit triggering LVD reset. During cold start-up on an initial $V_{EVRSB}$ ramp-up, RSTSTAT.STBYR is set. RSTSTAT.PORST bit implicitly set as LVD reset would trigger also warm PORST.	ON EVRSTAT.OVSB (SMU alarm) EVRSTAT.UVSB (SMU alarm) EVRMONSTAT2.ADCSB	2.97-5.50 V	External 5V or 3.3V EVR / Standby Supply to be provided
$V_{DDM}$	not available.	ON EVRSTAT.OVDDM (SMU alarm) EVRSTAT.UVDDM (SMU alarm) EVRMONSTAT2.ADCVDDM	2.97-5.50 V	External Supply to be provided
$V_{DDPD}$	ON.(via VDDPD POR detector) RSTSTAT.STBYR set if $V_{DDPD}$ drops below lowest voltage limit triggering LVD reset. RSTSTAT.PORST bit implicitly set as LVD reset would trigger also warm PORST.	ON EVRSTAT.OVPRE (SMU alarm) EVRSTAT.UVPRE (SMU alarm) EVRMONSTAT2.ADCPRE	1.125-1.375 V	Internal voltage not available on pin.

STANDBY system mode during supply modes a,d,e & h.

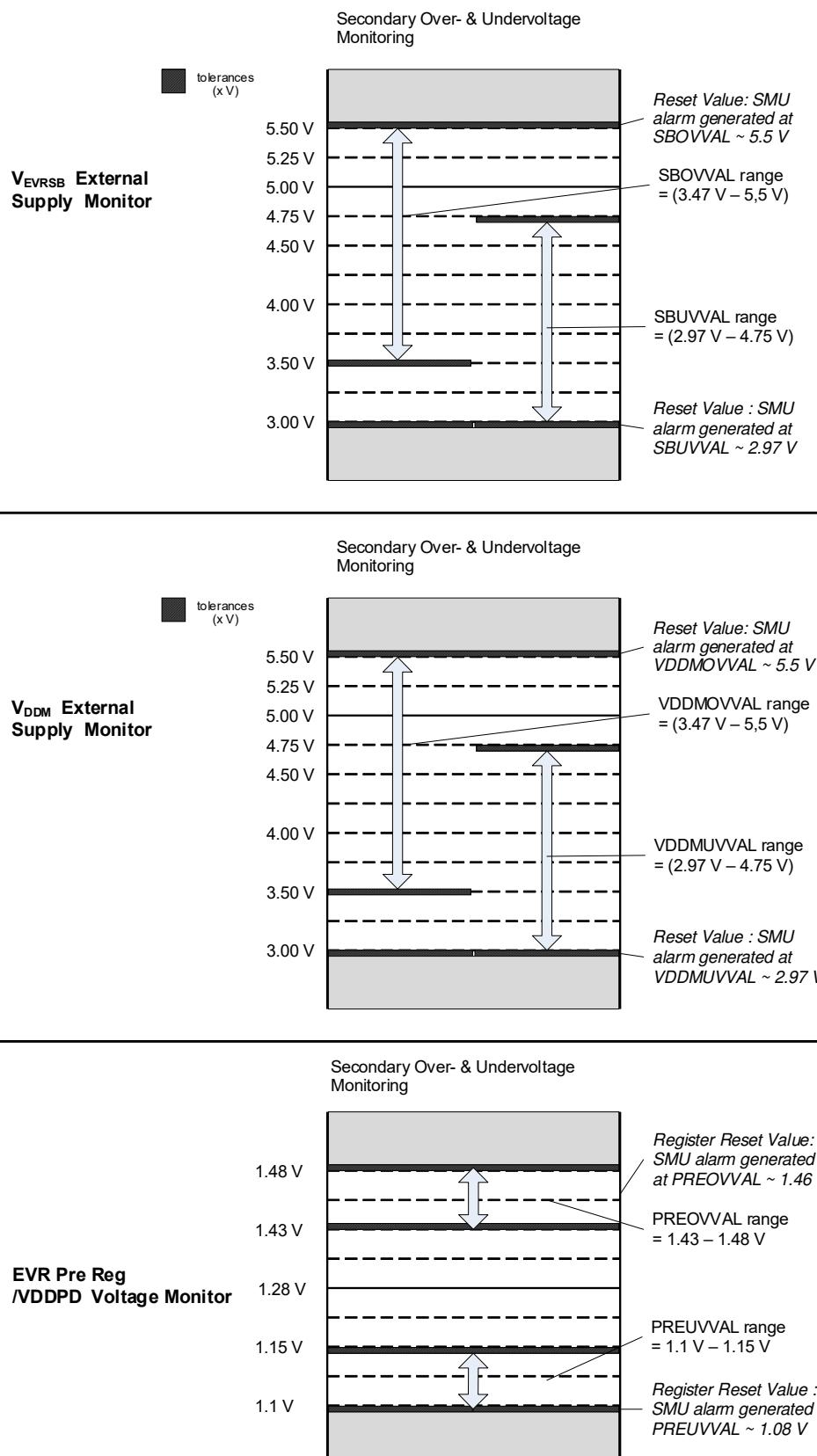
$V_{EXT}$	OFF/ON based on VEXTSTBYEN. RSTSTAT.STBYR set if $V_{EXT}$ drops below VLDRST5 voltage limit triggering LVD reset during Standby mode if VEXTSTBYEN = 0 & PWRWKEN = 0 is configured before Standby entry.  If Standby entry is triggered by power fail events; RSTSTAT.SWD, EVRC, EVR33 and RSTSTAT.PORST may be additionally set.	OFF	2.97-5.50 V	ON OFF if separate $V_{EVRSB}$ Standby supply used.
$V_{DDP3}$	OFF	OFF	0 V	OFF
$V_{DD}$	OFF	OFF	0 V	OFF
$V_{EVRSB}$	ON.(via VEVRSB detector) RSTSTAT.STBYR set if $V_{EVRSB}$ drops below VLDRSTSB voltage limit triggering LVD reset during Standby mode.	OFF	2.97-5.50 V	External Standby Supply to be provided
$V_{DDM}$	not available.	OFF	0-5.50 V	ON or OFF
$V_{DDPD}$	ON.(via VDDPD POR monitor) RSTSTAT.STBYR set if $V_{DDPD}$ drops below lowest voltage limit triggering LVD reset.	OFF	1.125-1.375 V	Internal voltage not available on pin.

## Power Management System (PMS)



**Figure 112 Voltage Monitoring - VEXT, VDDP3 & VDD**

## Power Management System (PMS)



Monitoring tolerances and levels are specified in datasheet section EVR : Supply monitoring

**Figure 113 Voltage Monitoring - VEVRSB, VDDM & VDDPD**

## Power Management System (PMS)

### 11.2.2.5.3 Power Built In Self Test at Start-up (PBIST)

A Power Built-In-Self-Test (PBIST) at start-up allows the testing of supply levels, power functions and voltage monitors before cold PORST reset release.

The internal EVRPR Pre-regulator VDDPD voltage based on the primary low power bandgap (PLPBG) is tested using secondary monitor ADC against the secondary bandgap (SHPBG) at supply ramp-up. This allows to monitor the bandgap voltages against each other during start-up and the device continue to remain in reset state till the test has passed. During runtime, bandgap monitoring is realised by VDDPD monitoring using secondary monitor ADC and alarm is raised to SMU in case of VDDPD over and under-voltage event.

VEVRSB and VEXT voltage levels are checked using secondary monitor ADC before starting the regulators in PBIST state. In case the voltages are not within the limits, the device reset state is not deasserted. Furthermore, the PBIST test is passed and reset state is deasserted when VDDM supply voltage is above 500 mV.

After EVRC and EVR33 regulators are ramped up, additional overvoltage and undervoltage checks are carried out for VEVRSB ( $5,84V / 2,75V \pm 5\%$ ), VEXT ( $5,84V / 2,75V \pm 5\%$ ), VDDP3 ( $3,81V / 2,0V \pm 5\%$ ), VDD ( $1,46V / 1,0V \pm 5\%$ ) and VDDPD ( $1,46V / 1,0V \pm 5\%$ ) rails before cold PORST reset release in PBIST2 state. The limits are the default reset values of **EVROVMON**, **EVROVMON2**, **EVRUVMON** and **EVRUVMON2** registers.

### 11.2.2.5.4 Secondary Monitor and Standby SMU Built in Self Test (MONBIST)

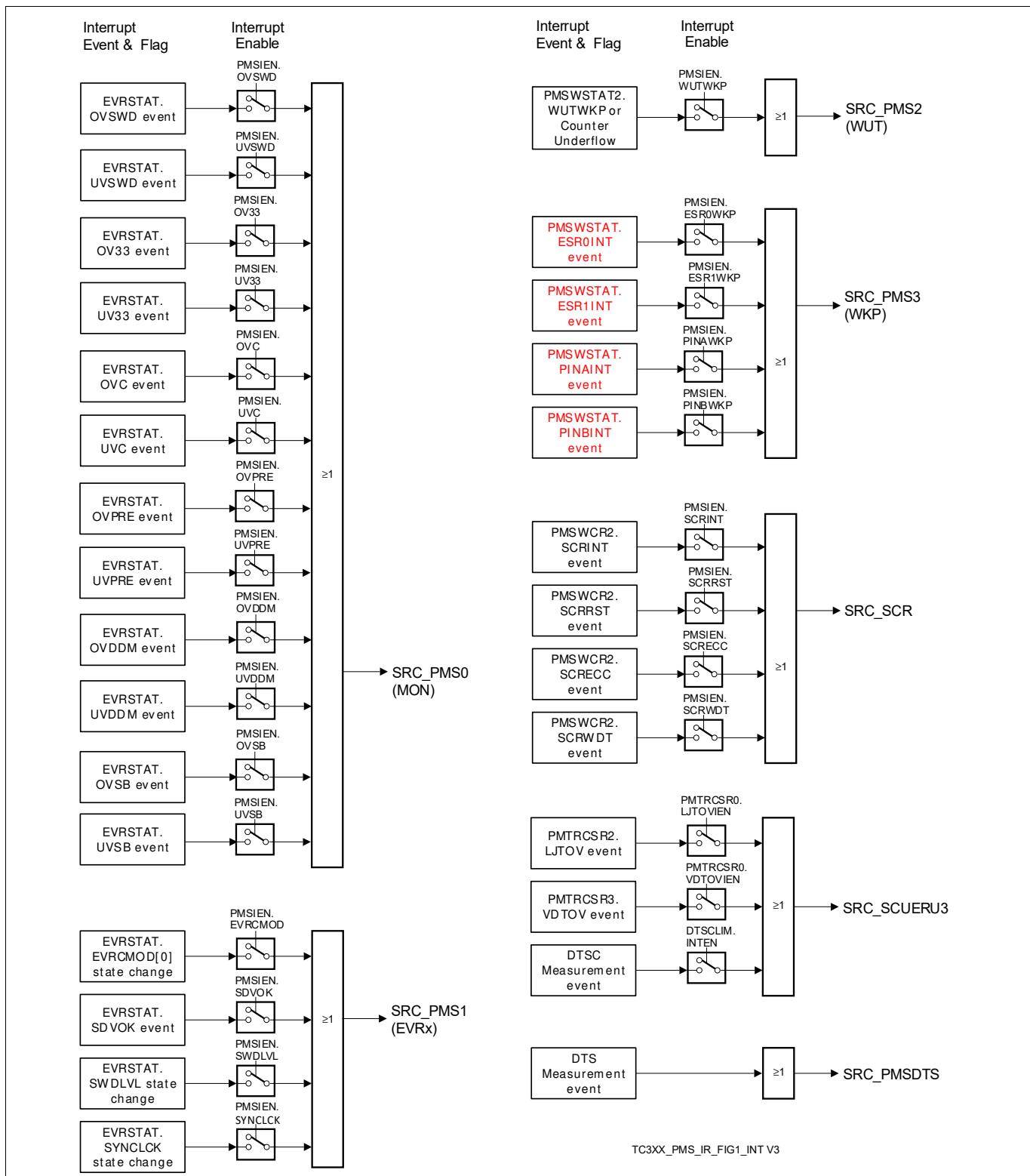
After reset release, MONBIST for the secondary monitors and alarm generation path may be carried out by user software. Secondary Monitor BIST ensures a higher latent fault coverage for the secondary monitors and the associated alarm and error pin fault logic routed to the Standby SMU. The MONBIST can be triggered during start-up via MONBISTCTRL.TSTEN register bit in Standby SMU module. During ongoing MONBIST, PMS SFF test shall not be triggered. MONBIST test takes less than 25 us execution time. The procedure is as follows :

- The Standby SMU shall be enabled via SMUEN register bitfield for MONBIST functionality.
- The MONBISTCTRL.TSTCLR bit shall be set foremost to clear all the flags and reset the test logic. This clears TSTEN, TSTRUN, TSTDONE, TSTOK, SMUERR and PMSERR bits.
- **EVRMONFILT** is set to 0x20000000 to clear the filter and to activate 1 x spike filter.
- **EVRMONCTRL** is set to 0xa5a5a5 to activate Over-voltage and Under-voltage alarms.
- The corresponding Over-voltage and Under-voltage interrupts are disabled by clearing **PMSIEN**.OVx/UVx register bit fields.
- FSP reaction on alarms are disabled by setting AGFSP.FEx to 0.
- CMD.FSP0EN and CMD.FSP1EN configuration bits are cleared to avoid spurious Error pin activation during MONBIST.
- CMD.ASCE is set to ensure that all pending alarms are cleared in AGx registers.
- **EVRMONFILT** is set back to 0x00000000 before enabling MONBIST to ensure alarm propagation.
- Consequently the MONBIST is enabled via MONBISTCTRL.TSTEN register bit.
- The MONBISTSTAT.TSTRUN register bit is set to indicate an ongoing test by MONBIST logic.
- Once the test is completed, MONBISTSTAT.TSTDONE bit is set and MONBISTSTAT.TSTRUN bit is cleared.
- The MONBISTSTAT.TSTOK bit indicates that the test was successfully completed.
- The MONBISTSTAT.SMUERR and MONBISTSTAT.PMSERR bits indicate that errors were detected during the MONBIST.
- FSPERR bit shall be cleared after MONBIST before enabling FSP reaction. If alarms happened during MONBIST, status registers may be updated and shall be cleared before Standby SMU initialization.TSTEN bit is cleared at the end of MONBIST.

## Power Management System (PMS)

### 11.2.2.6 Interrupts

Following events may be configured to lead to interrupts routed to Interrupt Router in Normal Run and Sleep System modes. If enabled by the related interrupt enable bit in register **PMSIEN**, an interrupt pulse can be generated on one of the service request outputs (SRC\_PMS0, SRC\_PMS1, SRC\_PMS2, SRC\_PMS3, SRC\_SCR, SRC\_SCUERU3). Interrupts are forwarded from PMS to IR module within 4 fspb clock cycles after the occurrence of the event.



**Figure 114 Interrupt Sources and Events**

## Power Management System (PMS)

### 11.2.2.7 OCDS Trigger Bus (OTGB) Interface

#### PMS OTGB Features

- Voltage signals and ADC outputs
  - Primary VDD, VDDP3 and VEXT voltage monitor outputs
  - Primary EVRC (SMPS) core voltage feedback ADC output
  - Secondary VDD, VDDP3 and VEXT voltage monitor outputs
  - EVRPR / VDDPD voltage monitor output
  - VEVRSB Standby supply voltage monitor output
  - VDDM ADC supply voltage monitor output
  - DTS temperature output
- EVR control outputs
  - EVR33 regulator DAC control output
  - EVRC switching control output routed to external MOSFETs
  - EVRC Regulator output and internal signals
  - Wake-up timer count
  - PMS and SCR register interface signals

The PMS module has two 16 bit ([Table 293](#)) trigger sets which are selected with the **OTSS** register. The trigger sets can be arbitrarily mapped to OTGB0/1 busses. Refer OCDS chapter for more details.

**Table 293 PMS Trigger Sets**

Trigger Set	Details
<a href="#">TS16_ADCMON Monitor Trigger Set</a>	<a href="#">Table 294</a>
<a href="#">TS16_EVRCON Control Trigger Set</a>	<a href="#">Table 295</a>

The PMS trigger signals relate to the 100 MHz internal back-up clock, which can be different to the OTGB/OTGM clock. It should be taken care that the triggers and associated signals are synchronised to SPB clock domain.

#### 11.2.2.7.1 ADC Monitor and Voltage Trigger Sets

ADC Monitor Trigger Sets consist of the important voltage signals measured by various PMS ADC monitors. The multiplexer allows to map arbitrary and different signal groups to the high and the low byte of a 16 bit Trigger Set. In addition it is possible to use one or two 16 bit Trigger Sets with this flexibility. All this is controlled with **OTSCO**.

## Power Management System (PMS)

**Table 294 TS16\_ADCMON Monitor Trigger Set**

Bits	Name	Description
[7:0]	SG0	8 bit Analog output from selected Analog monitors
		PRADCCV Primary Core / VDD voltage monitor output
		PRADC33V Primary VDDP3 voltage monitor output
		PRADCSWDV Primary VEXT voltage monitor output
		PRADCFBCV Primary EVRC SMPS core voltage feedback output
		SECADCCV Secondary Core / VDD voltage monitor output
		SECADC33V Secondary VDDP3 voltage monitor output
		SECADCSWDV Secondary VEXT voltage monitor output
		SECADCPRE EVRPR / VDDPD voltage monitor output
		SECADCSB VEVRSB standby voltage monitor output
		SECADCVDDM VDDM ADC voltage monitor output
		DTSRESULTL DTS Temperature output [7:0]
		DTSRESULTH DTS Temperature output [11:8]
[15:8]	SG1	Independent selection with same options as for Bits [7:0]

### 11.2.2.7.2 EVR Control output Trigger Sets

EVRCON Control Trigger Sets consist of the important control outputs of various regulators in PMS subsystem. All this is controlled with **OTSC1**.

**Table 295 TS16\_EVRCON Control Trigger Set**

Bits	Name	Description
[15:0]	EVR33OUT	EVR33 regulator DAC control output
	EVRCDPWM	EVRC digital PWM switching output to the external MOSFET
	EVRCOUT	Array of EVRC regulator signals from the SMPS module selected via DMOND multiplexer.
	WUTCNT	Wake-up timer count ([23:15] reduced to 15th bit)
	TCINT [7:0] SCRINT [15:8]	PMS and SCR output and input bus interface

## Power Management System (PMS)

### 11.2.3 Power Management

#### 11.2.3.1 Power Management Overview

The Power Management scheme allows activation of power down modes so that the system operates with the minimum required power for the corresponding application state. A progressive reduction in power consumption is achieved by invoking Idle, Sleep or Standby modes respectively. The Idle mode is specific to each individual CPU where as Sleep and Standby modes influence the complete system.

As shown in [Table 296](#), there are two power modes available for each CPU:

- CPU Run Mode
- CPU Idle Mode

**Table 296 CPU Power Management**

Mode	Description
<b>Run Mode</b>	The CPU clock is active and code is being executed.
<b>Idle Mode</b>	<p>CPU may enter Idle Mode on following events:</p> <ul style="list-style-type: none"> <li>• On a SW Idle request issued by setting register bits PMCSR<sub>x</sub>.REQSLP = 01<sub>B</sub> when CPU has no active tasks to perform.</li> <li>• On a SW Idle request (PMCSR<sub>y</sub>.REQSLP = 01<sub>B</sub>) issued by another CPU.</li> </ul> <p>The CPU code execution is halted and CPU clock is disabled in Idle state. The peripherals continue to remain active. CPU RAM memories (PSPR / DSPr / DLMU) are accessible to other bus masters and peripherals.</p> <p>CPU may exit Idle mode on following events:</p> <ul style="list-style-type: none"> <li>• When an interrupt occurs on a CPU returning the CPU to Run Mode.</li> <li>• When a trap occurs like an NMI trap event.</li> <li>• When the CPU watchdog or Safety watchdog timer overflow events trigger an SMU alarm in turn leading to a CPU interrupt.</li> <li>• When a MSB bit wrap of the CPU Watchdog counter takes place.</li> <li>• When a Application reset, System reset or any higher reset occurs.</li> <li>• On a SW Run request (PMCSR<sub>x</sub>.REQSLP = 00<sub>B</sub>) issued by another CPU.</li> </ul>

As shown in [Table 297](#), there are three main power modes available for the system:

- System Run Mode
- System Sleep Mode
- System Standby Mode

Furthermore, flexible reduction of power consumption is possible through following measures:

- Reduction of individual CPU power consumption by means of CPU clock scaling.
- Disabling the module clock by setting bit DISR in module CLC register if the module need not be active at the current point of time.
- Reducing the system frequency without changing individual peripheral clocks.
- Reducing individual peripheral clock frequency without changing system clock frequency. Main peripherals are provided with independent clocks separate from main system SRI and SPB clocks.

## Power Management System (PMS)

**Table 297 System Power Management**

Mode	Description
<b>Run Mode</b>	At least one master CPU has not requested Sleep Mode or Standby mode and is in Run mode. All peripheral modules are active.
<b>Sleep Mode</b>	<p>System may enter Sleep Mode on following events:</p> <ul style="list-style-type: none"> <li>• On a SW Sleep request issued by setting PMCSR<sub>x</sub>.REQSLP = 10<sub>B</sub> by the master CPU. CPU code execution is halted and CPU Idle state is entered. Peripherals are set into sleep state if so configured in the respective CLCx.EDIS bit. Ports retain their earlier programmed state.</li> </ul> <p>System may exit Sleep mode on following events:</p> <ul style="list-style-type: none"> <li>• When an interrupt or trap occurs on the master CPU.</li> <li>• When an NMI trap event takes place.</li> <li>• When the CPU watchdog or Safety watchdog timer overflow events trigger an SMU alarm leading in turn to a master CPU interrupt.</li> <li>• When a MSB bit wrap of master CPU Watchdog counter takes place.</li> <li>• When an Application reset, System reset or any higher reset occurs.</li> </ul>

## Power Management System (PMS)

**Table 297 System Power Management (cont'd)**

Mode	Description
Standby Mode (VEVRSB and VEXT supplied)	<p>System may enter Standby Mode on following events if so configured:</p> <ul style="list-style-type: none"> <li>• Standby entry on a SW Standby request issued by setting PMCSR<sub>x</sub>.REQSLP= 11<sub>B</sub> by the master CPU.</li> <li>• Standby entry on an ESR1 (NMI) assertion event. ESR1 (NMI) function doesn't require involvement of interrupt subsystem if configured as the standby entry trigger.</li> </ul> <p>The Standby domain constituting the Standby RAM, the 8 bit Standby Controller, shared ports and the wake-up unit remain actively supplied. The power to the rest of the chip is completely switched off. VEXT and VEVRSB rails remain supplied during Standby mode. VDDP3 and VDD supply rails are switched off.</p> <p>System may exit Standby mode on following events:</p> <ul style="list-style-type: none"> <li>• when a wake-up edge is detected on selected pins / ESR1.</li> <li>• when a wake-up request is issued by the 8 bit Standby Controller (SCR).</li> <li>• when a wake-up request is issued by Wake-up timer.</li> <li>• when PORST pin is asserted.</li> </ul>
Standby Mode (Only VEVRSB supplied)	<p>System may enter Standby Mode on following events if so configured:</p> <ul style="list-style-type: none"> <li>• Standby entry on a secondary under-voltage event during VEXT supply ramp-down.</li> <li>• Standby entry on an ESR1 (NMI) assertion event.</li> <li>• Standby entry on a SW Standby request issued by setting PMCSR<sub>x</sub>.REQSLP= 11<sub>B</sub> by the master CPU.</li> </ul> <p>The Standby domain constituting the Standby RAM and the 8 bit Standby Controller and Ports 33 and 34 remain actively supplied. The power to the rest of the chip is completely switched off. Only VEVRSB standby supply pin remain powered during Standby mode. VEXT, VDDP3 and VDD supply rails are switched off. SCR, WUT, Standby RAM supply maybe active or inactive during Standby mode.</p> <p>System may exit Standby mode on following event:</p> <ul style="list-style-type: none"> <li>• when VEXT supply ramps up</li> <li>• when a wake-up request is issued by SCR and VEXT is available.</li> <li>• when a wake-up request is issued by Wake-up timer and VEXT is available.</li> <li>• when a wake-up edge is detected on Pin B.</li> </ul>

## Power Management System (PMS)

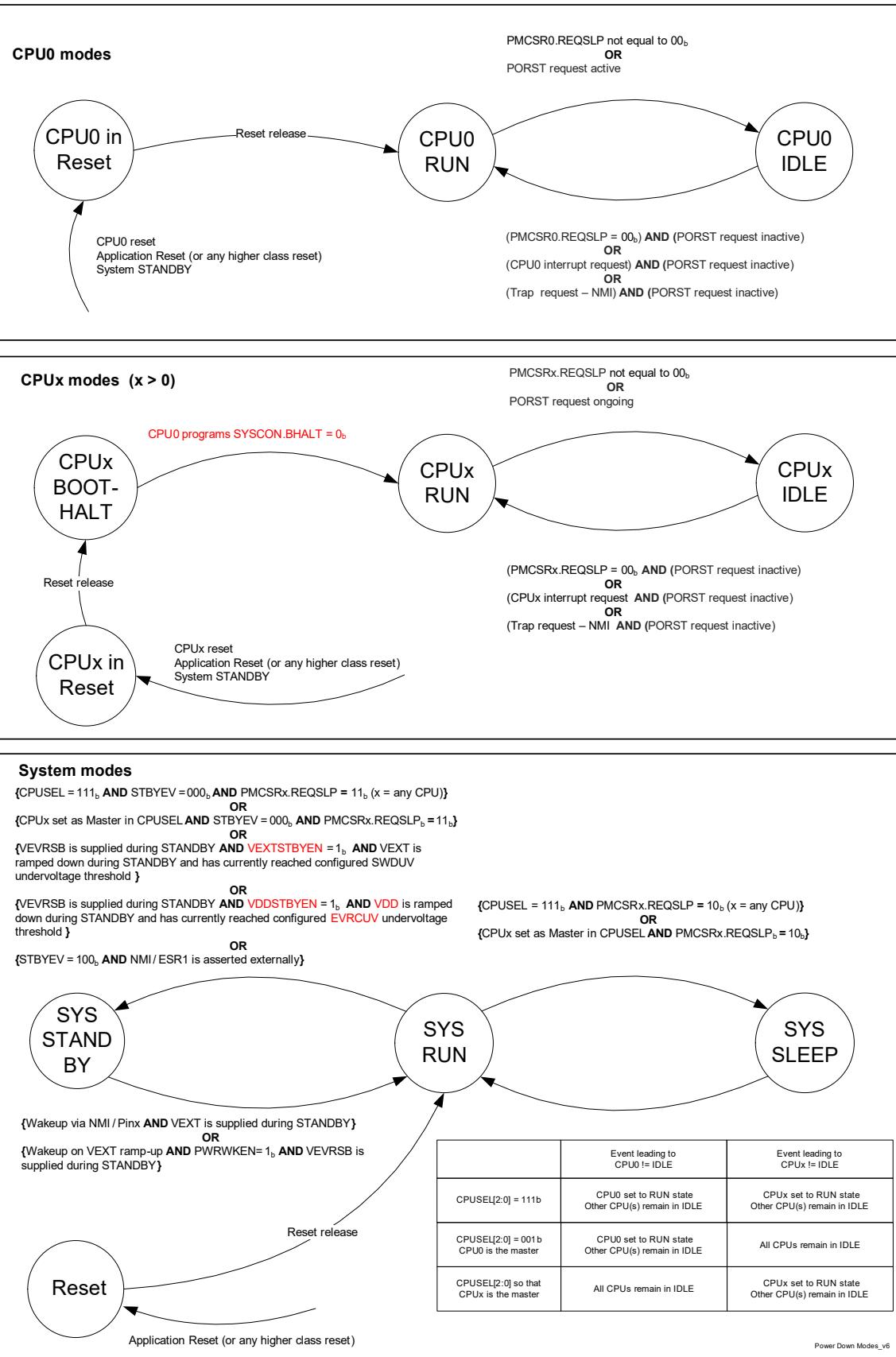


Figure 115 Power down modes and transitions

## Power Management System (PMS)

### 11.2.3.2 Idle Mode

In case there are no active tasks to perform, a CPU may be requested to enter Idle mode during runtime by writing to the respective PMCSR<sub>x</sub> register and setting the bit field REQSLP = 01<sub>B</sub>.

#### 11.2.3.2.1 Entering Idle Mode :

Following events can invoke a CPU<sub>x</sub> Idle request

- CPU<sub>x</sub> setting itself in Idle by writing its own PMCSR<sub>x</sub> register: The respective PMCSR<sub>x</sub> register shall be accessed by setting CPU<sub>x</sub> ENDINIT = 0<sub>B</sub> and consequently writing the bit field REQSLP = 01<sub>B</sub>. The Idle transition takes place only when CPU<sub>x</sub> ENDINIT = 1<sub>B</sub> is set back again. This ensures that a CPU<sub>x</sub> does not enter Idle mode when it's WDTx is in Time-Out mode and ENDINIT<sub>x</sub> = 0<sub>B</sub> to avoid wake-up on a consequent WDT time-out. Safety ENDINIT mechanism shall not be used by a CPU to set itself into Idle to avoid wake-up on a Safety WDT time-out. Idle mode may also be simultaneously triggered for additional CPUs based on SCU\_PMSWCR1.CPUIDLSEL configuration.
- Masters except CPU<sub>x</sub> setting CPU<sub>x</sub> into Idle (e.g.- CPU[y]): The PMCSR<sub>x</sub> register shall be accessed by such masters by setting Safety ENDINIT = 0<sub>B</sub>. The Idle request is issued immediately on setting PMCSR<sub>x</sub>.REQSLP = 01<sub>B</sub>. The device no longer waits for Safety ENDINIT = 1<sub>B</sub> to be set back to trigger Idle transition. It need to be taken care to grant access via ACCEN register as required by application.

The CPU watchdog may be disabled or slowed down by reprogramming the timers before triggering Idle request via Software. On an Idle request, the CPU finishes its current operations and sends an acknowledge signal back to the Power Management unit. It then enters an inactive state in which the CPU clocks and the respective DMI and PMI memory units are shut off. It is recommended to reduce respective CPU clocks via CPU<sub>x</sub>DIV register bit field before issuing Idle request.

#### 11.2.3.2.2 State during Idle mode

During Idle Mode, memory accesses to the DMI, PMI and DLMU from other bus masters cause these units to wake-up automatically to handle these transactions. When memory transactions are complete, the DMI, PMI and DLMU return to Idle state again. Once Idle Mode is entered, the state is reflected in PMCSR<sub>x</sub>.PMST status bits.

**Table 298 CPU[x] Idle Mode Entry Sequence, Behavior and Status Indication**

Condition	CPU[x] writes PMCSR[x].REQSLP = 01 <sub>B</sub>	Masters except CPU <sub>x</sub> (e.g.- CPU[y]) writes PMCSR[x].REQSLP = 01 <sub>B</sub>	CPU[y] writes PMCSR[y].REQSLP = 01 <sub>B</sub>
CPU[x] enters Idle Mode	A CPU[x] should be able to set itself into Idle.  CE[x] = 0 <sub>B</sub> SE = 0 <sub>B</sub> or 1 <sub>B</sub> PMCSR[x].REQSLP= 01 <sub>B</sub> CPU[x] Idle Entry happens when CE[x] = 1 <sub>B</sub> is set. If CE[x] = 1 <sub>B</sub> during PMCSR[x] write; FPI error issued and request is not taken.	A CPU[y] or (other masters except CPU[x]) should be able to set another CPU[x] into Idle if it has SE rights.  SE = 0 <sub>B</sub> CE[x] = 0 <sub>B</sub> or 1 <sub>B</sub> PMCSR[x].REQSLP= 01 <sub>B</sub> CPU[x] Idle Entry happens immediately. If SE = 1 <sub>B</sub> during PMCSR[x] write; FPI error issued and request is not taken.	CPUIDLSEL = y+1 is already set by a CPU having SE rights before.  CE[y] = 0 <sub>B</sub> SE = 0 <sub>B</sub> or 1 <sub>B</sub> PMCSR[y].REQSLP= 01 <sub>B</sub> All CPUs go into IDLE. when CE[y] = 1 <sub>B</sub> is set. If CE[y] = 1 <sub>B</sub> during PMCSR[y] write; FPI error issued and request is not taken.

## Power Management System (PMS)

**Table 298 CPU[x] Idle Mode Entry Sequence, Behavior and Status Indication (cont'd)**

Condition	CPU[x] writes PMCSR[x].REQSLP = 01 <sub>B</sub>	Masters except CPUx (e.g.- CPU[y]) writes PMCSR[x].REQSLP = 01 <sub>B</sub>	CPU[y] writes PMCSR[y].REQSLP = 01 <sub>B</sub>
CPU[x] during Idle Mode	PMCSR[x].REQSLP= 01 <sub>B</sub> PMCSR[x].PMST= 011 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>	PMCSR[x].REQSLP= 01 <sub>B</sub> PMCSR[x].PMST= 011 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>	PMCSR[ALL].REQSLP= 01 <sub>B</sub> PMCSR[ALL].PMST= 011 <sub>B</sub> PMSTAT0.CPU[ALL]&LS= 0 <sub>B</sub>
CPU[x] exits Idle mode	PMCSR[x].REQSLP= 00 <sub>B</sub> PMCSR[x].PMST= 001 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>	PMCSR[x].REQSLP= 00 <sub>B</sub> PMCSR[x].PMST= 001 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>	PMCSR[x].REQSLP= 00 <sub>B</sub> PMCSR[x].PMST= 001 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>

### 11.2.3.2.3 Exiting Idle mode

In Idle mode, the CPU will return to Run mode in response to the following wake-up events:

- An interrupt / trap received from an interrupt / trap source mapped to the CPU.
- An NMI trap request is received to wake-up the corresponding CPUs.
- A MSB bit wrap of the corresponding CPU Watchdog counter occurs.
- Setting the register bits PMCSR<sub>x</sub>.REQSLP = 00<sub>B</sub> to set the CPU<sub>x</sub> into Run mode.

The system enters reset state on an Application, System reset or any higher reset. If it is woken by a watchdog timer overflow event routed via the SMU to the CPU or by an NMI or by an interrupt, the CPU will immediately vector to the appropriate interrupt / trap handler.

CPU module reset will not result in exit from Idle mode if it was already in Idle state before. An explicit wake-up event has to happen before CPU is in run state again.

### 11.2.3.3 Sleep Mode

Sleep mode allows a progressive reduction of power consumption by gating the clocks of selected peripherals and keeping bare minimum modules active at their minimum clock frequencies during the Sleep state. Sleep mode maybe used to cater to Pretended Networking or ECU Degradation requirements. The clocks to a module maybe disabled individually using the respective CLCx.DISR register bits. Alternatively the clocks to selected peripherals may be simultaneously gated on a common sleep request if respective CLCx.EDIS register bits are cleared. The power consumption during Sleep state is predominantly dominated by the device leakage as power to the modules in core domain are not switched off. The dynamic core current component is reduced to the minimum as most of the module clocks are gated.

#### 11.2.3.3.1 Entering Sleep Mode

System may be requested to enter Sleep mode via software by master CPU by writing to the CPU's PMCSR<sub>x</sub> register and setting the bit field PMCSR<sub>x</sub>.REQSLP = 10<sub>B</sub>.

An example sequence for Sleep mode is enumerated below :

- The CLCx.EDIS register bit shall be cleared for all peripherals intended to be inactive in Sleep mode.
- All CPUs except the master CPU may be put into IDLE state. The respective watchdogs may be disabled or re-configured for slower modes. This allows to sequence the CPU load jumps before going into sleep mode
- Master CPU code execution maybe switched from Flash to PSPR RAM if required. Flash module may be explicitly set into Sleep state.
- The analog modules EVADC and EDSADC maybe switched off if not required to be active in Sleep state.
- It should be ensured to select the individual clocks from Clock Control Unit for peripherals which need to remain active during Sleep mode as shown in **Table 299**. Certain communication and timer peripherals have

## Power Management System (PMS)

clocks independent from the system frequencies, namely SRI and SPB clocks, to allow the possibility to bypass the system PLL. In such cases, the System PLL is switched into bypass mode and consequently the DCO would be switched off. Peripheral clock will continue to run clocking the modules active during sleep mode. The system clock frequencies, namely SRI and SPB clocks, maybe then reduced to the minimum possible values via the low power divider and / or Kx divider to reduce the current consumption. In some cases the respective peripherals may be clocked directly from external crystal / resonator depending on application.

- The interrupt control unit provides the infrastructure for wake-up from sleep state and therefore need to be kept active with a minimum SPB bus frequency. The respective module wakeup interrupts are routed to master CPU to wake-up on an interrupt event.
- Sleep Mode may be entered as a unanimous decision of all the CPUs when ALL of the PMCSR<sub>x</sub> registers in the system request it AND SCU\_PMSWCR1.CPUSEL = 111<sub>B</sub>. Sleep Mode may also be entered based on a singular decision of a master CPU based on the configuration of the CPUSEL register. The PMCSR<sub>x</sub> register shall be accessed by setting CPU<sub>x</sub> ENDINIT = 0<sub>B</sub>. The Sleep request is issued only after CPU<sub>x</sub> ENDINIT bit is set back again. Safety ENDINIT mechanism shall not be used by a CPU<sub>x</sub> to issue Sleep request. The master CPU watchdog may also be disabled or slowed down before issuing a Sleep request.

### 11.2.3.3.2 State during Sleep Mode

Sleep Mode is disabled for a unit if CLCx.EDIS bit is set. The sleep request is ignored in this case and the corresponding unit continues normal operation as intended. If CLCx.EDIS is cleared, the clock of the module is gated. CPU Idle state is entered for all the CPUs as described in the previous section. All ports retain their earlier programmed state. The current consumption during Sleep mode is documented in datasheet.

**Table 299 Module activity and configuration during Sleep mode**

Module active during Sleep mode	Module and Clock State during Sleep Mode
MCAN	<p>Peripheral System PLL active providing module clock (e.g. - f MOD = 20MHz - 40MHz).      Module may alternatively run on f OSC0 allowing also complete switch off of the Peripheral PLL.      Module FIFO and DMA allows autonomous handling of messages without involvement of CPU for a minimal amount of CAN messages.      System PLL may be switched into low power mode.      f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers (available only in B step).      System PLL may also be switched off and switched to Back-up clock depending on application (available only in B step).      Wake-up on CAN wake-up message identifier via CAN interrupt.</p>
ASCLIN	<p>Peripheral PLL active providing module clock (e.g. - f MOD = 20MHz).      Module may alternatively run on f OSC0 allowing also switch off of the Peripheral PLL.      System PLL may be switched into low power mode.      f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers.      Module FIFO and DMA allows autonomous handling of messages without involvement of CPU for a minimal amount of LIN frames.      System PLL may also be switched off and switched to Back-up clock depending on application.      Wake-up on LIN wake-up frame via ASCLIN interrupt.</p>

## Power Management System (PMS)

**Table 299 Module activity and configuration during Sleep mode (cont'd)**

Module active during Sleep mode	Module and Clock State during Sleep Mode
GPT12	<p>Peripheral PLL is disabled.</p> <p>System PLL may be switched into low power mode.</p> <p>f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers depending on application. Module clock (e.g - f MOD ~1-2 MHz) is derived from f SPB clock.</p> <p>System PLL may also be switched off and switched to Back-up clock depending on application.</p> <p>Wake-up on timer overflow or capture event via GPT12 interrupt.</p>
CCU6	<p>Peripheral PLL is disabled.</p> <p>System PLL may be switched into low power mode.</p> <p>f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers depending on application. Module clock (e.g - f MOD ~1-2 MHz) is derived from f SPB clock.</p> <p>System PLL may also be switched off and switched to Back-up clock depending on application.</p> <p>Wake-up on timer overflow or capture event via CCU6 interrupt.</p>
QSPI	<p>Peripheral PLL active providing module clock (e.g - f MOD = 20MHz).</p> <p>System PLL may be switched into low power mode.</p> <p>f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers.</p> <p>Services external watchdog if required by application. A timer module maybe used to trigger DMA or fill the FIFO allowing autonomous handling of messages without involvement of CPU.</p> <p>Wake-up in case of fault diagnosis of external device via a QSPI interrupt.</p>
Ethernet MAC	<p>Ethernet PHY active and provides module clock (e.g - f MOD = 25MHz) to the asynchronous part to decode the magic packet. Wake-up on magic packet via ETH interrupt.</p> <p>Alternatively PHY may trigger a wakeup directly via GPIO edge capture.</p> <p>Peripheral PLL may be disabled.</p> <p>System PLL may be switched into low power mode.</p> <p>f SRI, f SPB &amp; f ETH clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers.</p> <p>System PLL may also be switched off and switched to Back-up clock depending on application.</p>
I2C	<p>Peripheral PLL active providing module clock (e.g - f MOD = 20MHz).</p> <p>System PLL may be switched into low power mode.</p> <p>f SRI and f SPB clocks are reduced to (e.g. - 5 MHz) via LPDIV and / or Kx dividers.</p> <p>External communication remains active.</p>
GTM	<p>Peripheral PLL is disabled.</p> <p>System PLL is active and provides the module clock (e.g - f MOD ~f SPB ~ 1-2 MHz).</p> <p>f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers.</p> <p>It is recommended to switch off the GTM module completely and use smaller timer modules like CCU6 / GPT12, STM or WUT during Sleep state to reduce power consumption.</p> <p>Wake-up on timer overflow or capture event via GTM interrupt.</p>
STM	<p>Peripheral PLL is disabled.</p> <p>System PLL is active and provides the module clock (e.g - f MOD ~f SPB ~ 1-2 MHz).</p> <p>f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers.</p> <p>Wake-up on timer overflow via STM interrupt.</p>

## Power Management System (PMS)

**Table 299 Module activity and configuration during Sleep mode (cont'd)**

Module active during Sleep mode	Module and Clock State during Sleep Mode
Pin Wake-up ESR1 (NMI)	Peripheral PLL is disabled. System PLL bypassed. f SRI, f SRI clocks reduced to (e.g. - ~1-2 MHz) via LPDIV / Kx dividers. System PLL may also be switched off and switched to Back-up clock depending on application. Wake-up on Edge / Level detection on pin routed to ERUx, ESR1 (NMI) or PORT module via SCU interrupts or polling Port registers on an active timer interrupt.
WUT	Peripheral PLL is disabled. System PLL bypassed. f SRI, f SRI clocks reduced to (e.g. - ~1-2 MHz) via LPDIV / Kx dividers. System PLL may also be switched off as module clock is derived from Back-up clock Wake-up on timer overflow via WUT interrupt.

### 11.2.3.3.3 Exiting Sleep Mode

The system will exit Sleep mode on any wake-up event that causes any master CPU to exit Idle Mode depending on CPUSEL configuration. Only the master CPU associated with the interrupt wake-up event would be set into Run mode (REQSLP = RUN, PMST = RUN). Other CPUs will remain in Idle (REQSLP = SLEEP, PMST = IDLE). An NMI trap event will wake-up the respective CPU as configured in TRAPDIS0 and TRAPDIS1 registers. A MSB bit wrap of the corresponding master CPU Watchdog counter would also wake-up the master CPU. The response of the CPU to being woken up from Sleep Mode is also the same as for Idle Mode. Peripheral units that have entered Sleep Mode will switch back to their selected Run Mode operation. Wake-up latency from Sleep mode depends mainly on the extent of clock ramp-up required after wake-up keeping the load jump constraints. If DCO or PLL is switched off, the wake-up latency would include the time to power and lock the PLL. The sequence after wake-up is dependent on the entry sequence and mainly constitutes ramping back the clock system, activating analog and Flash modules, switching from RAM to Flash execution and activating additional CPUs. The time taken between interrupt trigger availability until CPU has woken up and is executing next instruction is less than 3 SPB + 20 SRI clock cycles.

## Power Management System (PMS)

**Table 300 System Sleep Mode Entry Sequence, Behavior and Status Indication**

Condition	Master CPU[x] writes PMCSR[x].REQSLP = 10 <sub>B</sub>	CPU[y] writes PMCSR[x].REQSLP = 10 <sub>B</sub>	All CPU[y] writes respective PMCSR[y].REQSLP = 10 <sub>B</sub>
System enters Sleep Mode	A CPUx should be able to trigger SLEEP mode if CPUSEL = x+1 <sub>B</sub> is already set by a CPU having SE rights before.  CE[x] = 0 <sub>B</sub> SE = 0 <sub>B</sub> or 1 <sub>B</sub> PMCSR[x].REQSLP= 10 <sub>B</sub> System enters SLEEP mode when CE[x] = 1 <sub>B</sub> is set. If CE[x] = 1 <sub>B</sub> during PMCSR[x] write; FPI error issued and request is not taken.	CPU[y] is not authorised to trigger Sleep Mode and therefore this is an error case.  CPUx is configured to trigger SLEEP mode via CPUSEL = x+1 <sub>B</sub> . SE = 0 <sub>B</sub> CE[x] = 0 <sub>B</sub> or 1 <sub>B</sub> PMCSR[x].REQSLP= 10 <sub>B</sub>	CPUSEL = 111 <sub>B</sub> is already set by a CPU having SE rights before. CE[y] = 0 <sub>B</sub> PMCSR[y].REQSLP= 10 <sub>B</sub> System enters SLEEP mode if all CPUs have requested for SLEEP entry and respective CE[y] = 1 <sub>B</sub> is set.  If CE[y] = 1 <sub>B</sub> during PMCSR[y] write; FPI error issued and request is not taken.
System during Sleep Mode	PMCSR[x].REQSLP= 10 <sub>B</sub> PMCSR[x].PMST= 100 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>  PMCSR[y].REQSLP= 10 <sub>B</sub> PMCSR[y].PMST= 011 <sub>B</sub> PMSTAT0.CPU[y] & LS= 0 <sub>B</sub>	PMCSR[x].REQSLP= 10 <sub>B</sub> PMCSR[x].PMST= 011 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>  PMCSR[y].REQSLP= 01 <sub>B</sub> PMCSR[y].PMST= 011 <sub>B</sub> PMSTAT0.CPU[y] & LS= 0 <sub>B</sub>	PMCSR[y].REQSLP= 10 <sub>B</sub> PMCSR[y].PMST= 011 <sub>B</sub> PMSTAT0.CPU[y] & LS= 0 <sub>B</sub>
System during Sleep Exit	Wake-up on Master CPU PMCSR[x].REQSLP= 00 <sub>B</sub> PMCSR[x].PMST= 001 <sub>B</sub> PMSTAT0.CPU[x] & LS= 1 <sub>B</sub>  PMCSR[y].REQSLP= 10 <sub>B</sub> PMCSR[y].PMST= 011 <sub>B</sub> PMSTAT0.CPU[y] & LS= 0 <sub>B</sub>	System remains in RUN mode.	Wake-up event of respective CPU[x] PMCSR[x].REQSLP= 00 <sub>B</sub> PMCSR[x].PMST= 001 <sub>B</sub> PMSTAT0.CPU[x] & LS= 1 <sub>B</sub> Other CPU[y] remain in IDLE if not woken up PMCSR[y].REQSLP= 10 <sub>B</sub> PMCSR[y].PMST= 011 <sub>B</sub> PMSTAT0.CPU[y] & LS= 0 <sub>B</sub>

## Power Management System (PMS)

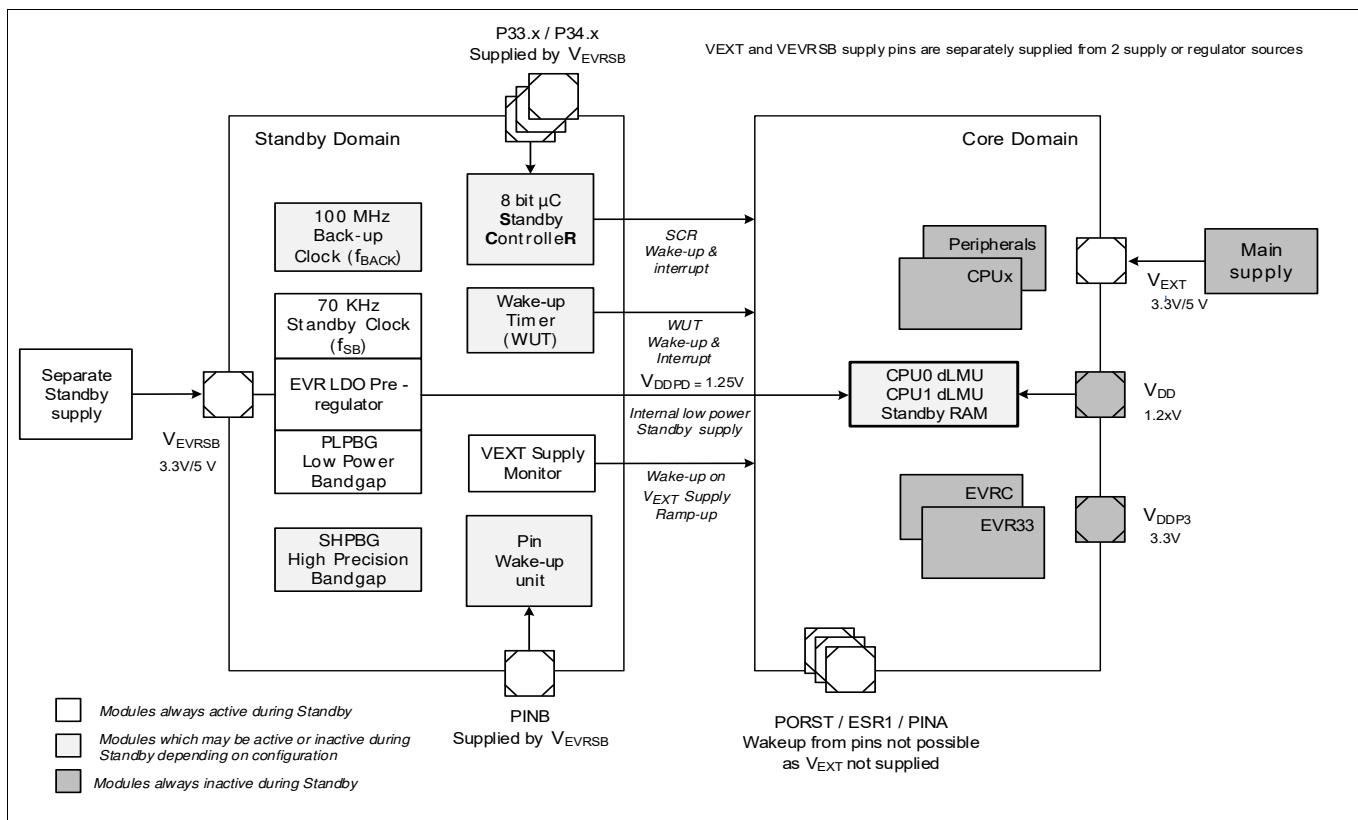
### 11.2.3.4 Standby Mode

The Standby domain constitutes the Standby RAM, the 8 bit Standby Controller, the Power Management unit, the Pin Wake-up unit, the Wake-up timer, the VEXT supply monitor and basic infrastructure components. The Standby domain is supplied by the EVRPR pre-regulator and is by default clocked by the 70 kHz internal low power clock source in Standby Mode. The 3.3V / 5V dedicated external Standby supply pin VEVRSB supplies the EVRPR pre-regulator and the Port domain P33.x / P34.x during the Standby mode when VEXT supply is switched off.

Following Standby topologies are supported with respective events which may trigger Standby mode entry and exit based on SCU\_PMSWCR1.STBYEV and **PMSWCR0.xWKEN** bits.

#### 11.2.3.4.1 Standby Mode with only VEVRSB domain supplied and VEXT domain switched off

As shown in **Figure 116**, only the Standby domain and Port domain P33.x/P34.x continue to be supplied by the separate VEVRSB supply pin during Standby mode. The main VEXT supply is switched off in Standby state. Consequently the rest of the PORT domain except P33.x/P34.x is devoid of supply.



**Figure 116 Standby domain supplied via a separate dedicated supply pin VEVRSB**

Standby Entry is triggered by following events :

- Standby entry on a secondary under-voltage event during VEXT supply ramp-down if configured in **PMSWCR0.VEXTSTBYEN** bits.
- Standby entry on SW request (PMCSR<sub>x</sub>.REQSLP = 11<sub>B</sub>) if configured via SCU\_PMSWCR1.STBYEV register bit field. The Standby request is issued only after CPU<sub>x</sub> ENDINIT bit is set back again. Safety ENDINIT mechanism shall not be used by a CPU<sub>x</sub> to issue Standby request. Standby entry on ESR1 (NMI) edge event if configured via SCU\_PMSWCR1.STBYEV register bit field.

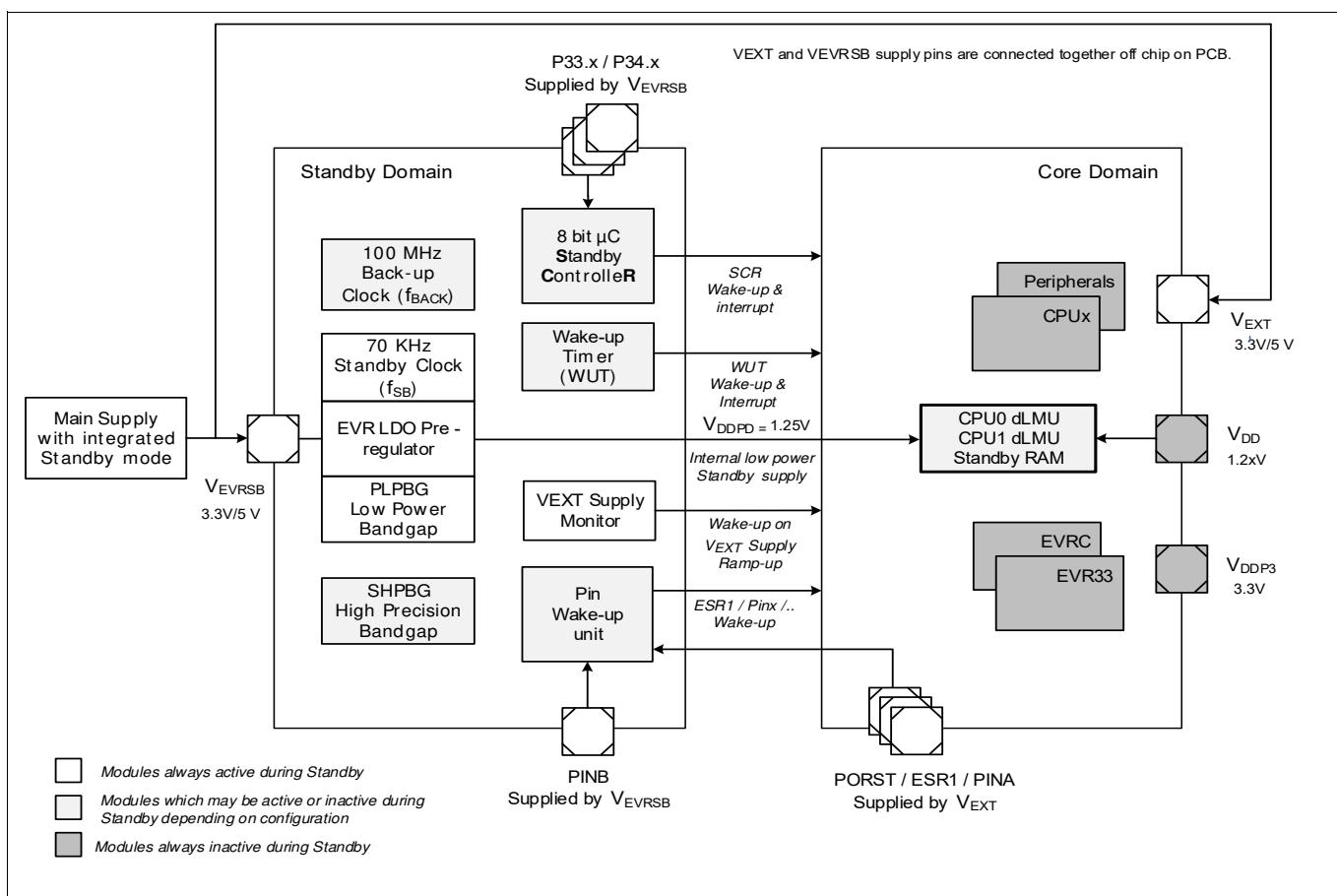
Standby Wake-up is triggered by following events after blanking filter time expiry :

## Power Management System (PMS)

- Wake-up is triggered when main VEXT supply ramps-up again if configured via **PMSWCRO.PWRWKEN** enable bit.
- Wake-up is triggered by Standby Controller if configured via **PMSWCRO.SCRWKEN** enable bit provided VEXT supply has already ramped-up before. Standby controller can also request for VEXT ramp-up to external regulator.
- Wake-up via Wake-up Timer if configured via **PMSWCRO.WUTWKEN** enable bit provided VEXT has already ramped-up before.
- Wake-up via Pin B if configured via **PMSWCRO.PINBWKEN** enable bit provided VEXT has already ramped-up before.
- It is to be noted that wake-up via PORST, Pin A, ESR0 & ESR1 pins which are in turn supplied by VEXT is not supported during STANDBY as VEXT is not supplied. Therefore it is required to disable the respective **PMSWCRO.xWKEN** bits.

### 11.2.3.4.2 Standby Mode with both VEXT and VEVRSB supplied via common supply rail.

As shown in **Figure 117**, the Standby domain and the complete Pad domain including P33.x/P34.x, PORST, ESRx and PINx continue to be supplied by (VEVRSB + VEXT) supply rail during Standby mode. This allows additional wake-up possibility via PORST, ESRx and PINx pins but at the cost of higher power consumption during Standby mode.



**Figure 117 Standby domain supplied via a common supply rail connected to both VEXT and VEVRSB**

Standby Wake-up is triggered by following events after blanking filter time expiry :

- Wake-up via NMI / Pinx: Wake-up on rising, falling or any edge of ESR1, Pin A or Pin B pins if configured via **PMSWCRO.ESRxWKEN** / **PINxWKEN** register bit fields.

## Power Management System (PMS)

- Wake-up is triggered by Standby ContolleR if configured via **PMSWCR0**.SCRWKEN enable bit.
- Wake-up via Wake-up Timer if configured via **PMSWCR0**.WUTWKEN enable bit.
- Wake-up via PORST pin if configured via **PMSWCR0**.PORSTWKEN enable bit.

Standby Entry is triggered by following events

- Standby entry on a secondary under-voltage event during VEXT supply ramp-down if configured in **PMSWCR0**.VEXTSTBYEN bits.
- Standby entry on SW request (PMCSR<sub>x</sub>.REQSLP = 11<sub>B</sub>) if configured via SCU\_PMSWCR1.STBYEV register bit field. The Standby request is issued only after CPU<sub>x</sub> ENDINIT bit is set back again. Safety ENDINIT mechanism shall not be used by a CPU<sub>x</sub> to issue Standby request.
- Standby entry on ESR1 (NMI) edge event if configured via SCU\_PMSWCR1.STBYEV register bit field.

### 11.2.3.4.3 Standby RAM

The Standby RAM constitutes ECC protected DLMU RAM of CPU0 (Block 0 and Block 1) and DLMU RAM of CPU1 (Block 0 and Block 1). The 32Kb Block 0 (lower half) is located at address \*0000H and 32Kb Block 1 (upper half) is located at address \*8000H of the respective address range of the corresponding CPUn DLMU RAM. The RAMs remain supplied during Standby mode if configured in **PMSWCR0**. STBYRAMSEL bits. On wake-up, the status which Standby RAMs remain supplied is reflected in **PMSWSTAT2**.STBYRAM bits. The initial 16 words from the start address of DLMU0/DLMU1 are not retained during standby mode as this memory region is used by start-up software.

The Standby RAM cell array is supplied by a separate supply pin (VEVRSB) during Standby state via the internal EVRPR Pre-Regulator. It shall be ensured that the external standby supply source continues to supply VEVRSB supply pin during Standby state with a supply between 2.6 V up to 5.5 V. Standby supply status is also monitored and indicated via RSTSTAT.STBYR bit which indicates EVRPR or VDDPD supply under-voltage LVD reset. It is to be taken care by the Start-up software after wake-up that Standby RAMs are not initialized if **PMSWSTAT2**.STBYRAM bits are set. Furthermore, if **PMSWCR0**.STBYRAMSEL bit is set to enable Standby RAM function and there was a VDD primary under-voltage (cold PORST) event, it is ensured that the Standby RAM supply is switched back to VDDPD supply rail. This ensures that RAM contents are not corrupted also during main VDD core supply loss.

### 11.2.3.4.4 VEXT Supply Monitor

If Standby mode is entered on a VEXT supply ramp down, the consequent wake-up on VEXT supply ramp up is triggered by the VEXT supply monitor activated by configuring **PMSWCR0**.PWRWKEN bit. The Standby request is issued by the secondary under-voltage monitor on crossing a voltage threshold as configured in **EVRUVMON** and **EVRMONCTRL** registers. Idle request acknowledge sequence issued to modules shall be deactivated on Standby entry by setting SCU\_PMSWCR1.IRADIS bit if VEXT supply is available. The EVR33 and EVRC regulators are switched off and Standby state is entered. Consequently VEXT and VDDM supplies may be ramped down, thus port and analog domains are also devoid of power. Wake-up is triggered when VEXT supply ramps up again and is detected by the VEXT supply monitor in the Standby domain. The detection time of the detector itself on reaching VLVDRST5 level is within 50 us. Nevertheless the complete time for start-up from Standby mode is quite the same as that for normal start-up and is documented tBP parameter in datasheet. VEXT wake-up is recognized as valid only after a minimum delay time has elapsed in Standby state as configured in **PMSWCR0**.BLNKFL register bits. This is to avoid spurious wake-up events owing to residual voltage on VEXT supply due to external buffer capacitors. After a successful wake-up, the register bit **PMSWSTAT**.PWRWKWP is set to indicate wake-up owing to a VEXT supply ramp-up and shall be cleared via **PMSWSTATCLR**.PWRWKPCCLR register bit.

### 11.2.3.4.5 Pin Wake-up Unit

External events may be mapped to ESRx / PINx pins in turn acting as wake-up signals for the system. In Run Mode, ESR1 pin may be used as fault or functional interface for external devices. In Standby Mode, an edge event on the

## Power Management System (PMS)

ESR1 pin may be configured to trigger wake-up of the main core domain via **PMSWCRO**.ESR1WKEN bit and is reflected in **PMSWSTAT**.ESR1WKEN status flag. It can be configured to trigger a wake-up on rising, falling or both edges via **PMSWCRO**.ESR1EDCON bit. The minimum pulse width of the external wakeup input signal without the digital filter activated shall be atleast 2 clock cycles. Glitches on ESR1 input are filtered out by activating the filter via **PMSWCRO**.ESR1DFEN bit. The reset behavior is documented in External Service Requests chapter in RCU chapter. Additional pins (PINA - P14.1 and PINB - P33.12) may likewise be configured to trigger wake-up via **PMSWCRO**.xEDCON, xDFEN & xWKEN bits. On wake-up, **PMSWSTAT**.ESR1WKP or PINxWKP event flags provide information as to the wake-up source. It should be taken care after wake-up to clear the event flags via **PMSWSTATCLR** register. In case new wake-up events are captured while **PMSWSTAT**.xWKP flags are still set, then **PMSWSTAT**.xOVRRUN flags are set to indicate an overrun state owing to consecutive un-serviced wake-up events.

### 11.2.3.4.6 Standby ContolleR (SCR) Interface

The 8 bit Standby controller (SCR) subsystem constitutes an XC800 core, 8KB XRAM memory, various timer modules, ADC comparator, various communication peripherals and up to 16 shared pins executing autonomous activity during Idle, Sleep and Standby modes. Various Standby functions and periodic monitoring tasks may be encapsulated in the SCR with minimal power consumption overheads.

The SCR is enabled via **PMSWCR4**.SCREN bit and the status is reflected in **PMSWSTAT**.SCR bit. If SCR is disabled via **PMSWCR4**.SCREN bit, it is ensured that SCR 100MHz clock request, pending requests from SCR wake-up sources and other SCR interfaces do not have any effect on the main system. After start-up, CPU0 programs the SCR via FPI interface and copies the code into the internal XRAM.

A reset may be issued to the SCR via **PMSWCR4**.SCRSTREQ register bit. Consequently **PMSWSTAT**.SCRST status register bit is flagged to indicate SCR reset. SCRRST register bit is cleared via **PMSWSTATCLR**.SCRSTCLR bit. **PMSWCR4**.SCRSTREQ register bit is cleared after reset has been issued. The SCR is reset in case of warm PORST assertion based on **PMSWCR4**.PORSTREQ register configuration reflected in PORST register bit during normal RUN and SLEEP modes. The SCR is not affected by an Application or System reset. After reset release, the firmware initializes the SCR subsystem based on the hardware configuration programmed in **PMSWCR4**.SCRCFG bits. In case of LVD reset, a complete power-on reset of SCR is carried out.

The 20 MHz stand-by clock source is the default SCR clock active in System Standby Mode enabling higher-performance of the SCR subsystem. The SCR clock source may be switched to the internal 20 MHz (derived from the 100 MHz back-up clock) clock source via SCRCLKSEL register bit thus enabling higher performance on the SCR subsystem. Fundamentally SCR is in control of its clock needs and may request a clock switch via CMCON.OSCPD register overruling the behavior configured in SCRCLKSEL register. A watchdog ensures that the clock received after the request is adequate for reliable operation.

SCR PORT module shares a part of the PORT (P33.0 - P33.7, P33.9 - P33.15 and P34.1) domain with the main port system which may be kept active during Standby mode. The SCR ports are supplied by VDDPD and VEVRSB standby supply. The control to these pins need to be allocated explicitly to the SCR via port configuration Pxx\_PCSR register. Unused wake-up pins may be configured as tristate in Standby Mode. Furthermore dedicated wake-up pins, namely ESR0, ESR1, PINA - P14.1 and PINB - P33.12 are also routed to the SCR subsystem to recognise wake-up edges on these pins. When SCREN = 0 is programmed, PINB wakeup is configured as explained in section [Section 11.2.3.4.5](#). Alternatively if SCREN=1 is programmed, PINB ownership is to be foremost transferred to SCR and SCR PINB Port configuration need to be set to input.

The SCR XRAM is accessible from the main domain via the FPI interface. Simultaneous access to XRAM via FPI interface and the SCR is arbitrated with SCR having default priority for XRAM access. In case of wake-up from Standby, it is ensured that SCR XRAM is not re-initialised.

An additional register interface with interrupt support using **PMSWCR2** register bit fields for exchange and facilitate status handshake between the two domains. The SCR can make a direct interrupt request to any CPUx by writing to register NMICON.SCRINTTC SCR register bit. An additional 8 bit information maybe written to

## Power Management System (PMS)

SCRINTEXCHG SCR register which is also transferred to **PMSWCR2**.SCRINT register bit field to decode the interrupt reason. The routing of the interrupt to the service request node need to be enabled via **PMSIEN**.SCRINT register bit.

Likewise any CPU may also trigger a direct interrupt request to the SCR by writing to **PMSWCR2**.TCINTREQ register bit. An additional 8 bit information maybe written to **PMSWCR2**.TCINT register which is likewise transferred to TCINTEXCHG SCR register bit field to decode the interrupt reason on SCR side.

Critical SCR errors / events like XRAM ECC errors, SCR watchdog overflow event and SCR internal reset need to be communicated back to the main core domain via **PMSWCR2**.SCRECC, SCRWD and SCRRST register bits. These events may additionally trigger internal SCR reset if configured in RSTST SCR register. The occurrence of SCRECC, SCRWD and SCRRST events may be routed to interrupt based on **PMSIEN**.SCRECC, SCRWD and SCRRST register bits.

Like-wise resets of the main system, namely application, system and power-on resets, are reflected in **PMSWCR2**.RST register bit and communicated to SCR register MRSTST.RST bit. Furthermore, SMURST is differentiated via **PMSWCR2**.SMURST and communicated to SCR register MRSTST.SMURST bit. Interrupt maybe generated in SCR subsystem when MRSTST register bits are set. The bits are cleared when SCR has latched the information.

During standby mode, the SAR secondary monitor ADC maybe used to carry out analog conversions of up to 4 analog inputs (P33.4, P33.5, P33.6 & P33.7) if requested by SCR. Refer SCR ADCOMP chapter for more details.

During a standby to run mode transition on a wake-up event, the P33 and P34 PCSR.SELx shadow register value retains the programmed value of PCSR.SELx value before standby entry. This is to ensure that SCR continues to have control over the respective P33 and P34 port pins during and after exit from Standby, though the Port register PCSR.SELx value is reset. Only on a consequent explicit write to the register after reset release will a new PCSR.SELx value be taken to switch the Port 33 and P34 control.

The SCR may wake-up the main core domain from Standby state if configured in SCRWKEN register bit. The enabling of SCR wake-up via SCRWKEN should be programmed when SCR is running at 20 MHz. A wake-up request is issued by the SCR SW via SCRWP bit in STDBYWKP register as documented in the SCR SCU chapter. On wake-up of the main core domain, SCRWP event flag is set which shall be cleared via SCRWPCLR register bit.

### 11.2.3.4.7 Wake-up Timer (WUT)

The Wake-up Timer is a basic low power counter which may be used to wake-up the system periodically from Standby mode. The timer may also be used during RUN, IDLE or SLEEP modes. The following list enumerates the salient features.

- 24 bit counter running on 70 kHz clock source with programmable reload value.
- 24 bit counter status register providing the current count value.
- Timer resolution of 70 kHz or (70 kHz / 2<sup>10</sup>) configured via a clock divider.
  - 14.3 us resolution : 14.3 us - 240 s range ± 60% default tolerance
  - 14.3 ms resolution : 14.3 ms - 2.7 days range ± 60% default tolerance
- 2 operating modes :
  - Auto Reload mode - WUT is started and stopped via Software. Automatic reload on counter underflow and triggers a system wake-up.
  - Standby Auto Stop mode - Counter starts counting down from reload value on Standby entry. Counter stops on underflow and triggers a system wake-up.
- Events on WUT counter underflow
  - Interrupt request on SRC\_PMSx (WUT) interrupt node on counter underflow during RUN, IDLE or SLEEP mode.
  - Wake-up trigger on counter underflow during STANDBY mode.

## Power Management System (PMS)

- Capture trigger on counter underflow to CCU60\_CC60IND, CCU61\_CC60IND and GTM (TIM 0.7) for trimming purpose.
- Over-run indication of consecutive un-serviced wake-up triggers

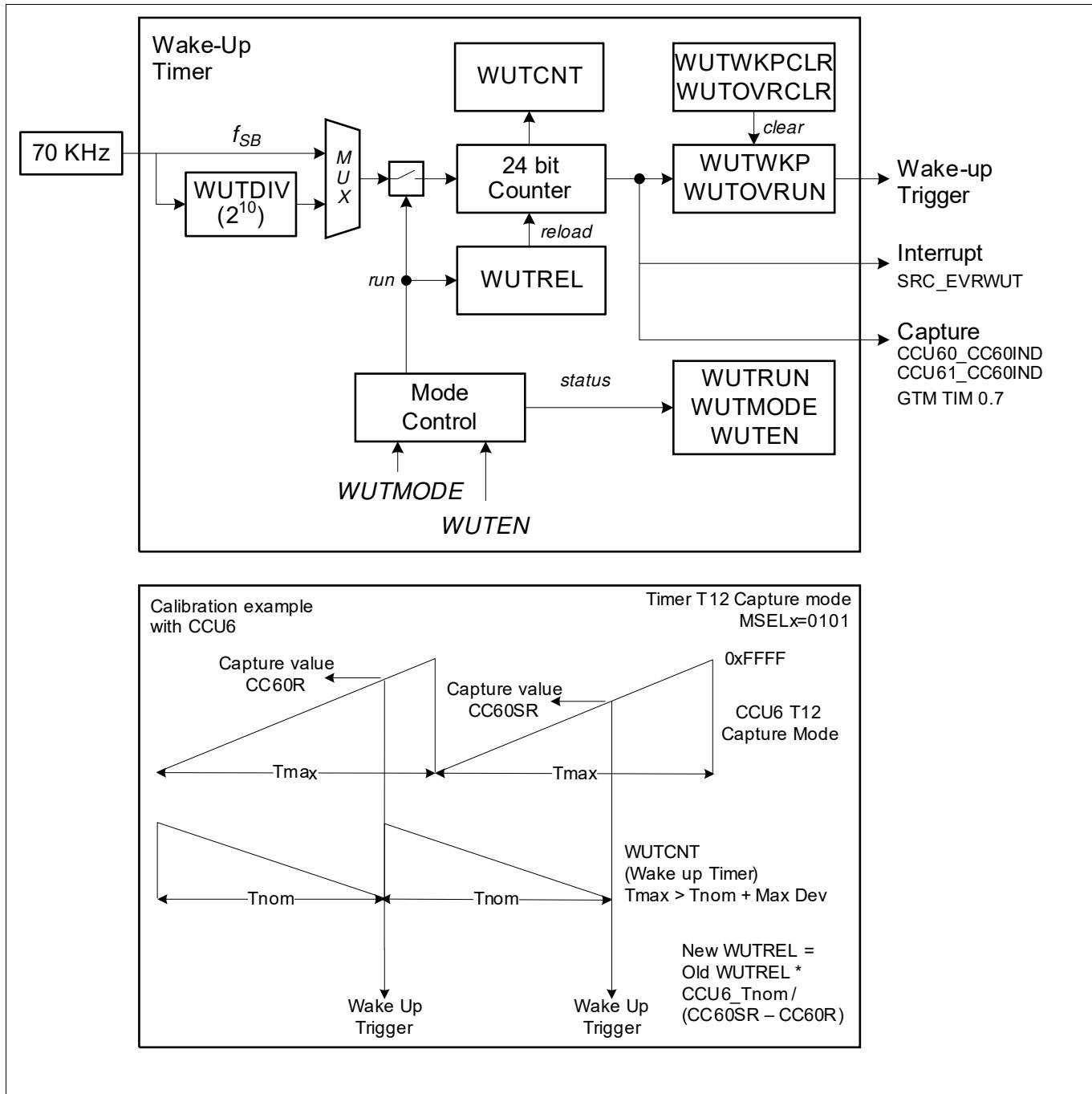


Figure 118 Wake-up Timer (WUT)

## Power Management System (PMS)

**Table 301 Wake-up Timer Operation and Modes**

WUTEN	WUTMODE	Mode Description
0B	XB	<p>WUT is disabled and counter is stopped.</p> <p><b>PMSWCR3.WUTREL</b> reload value may be updated.</p> <p><b>PMSWSTAT2.WUTCNT,WUTRUN,WUTWKP &amp; WUTOVR</b> flags read 0.</p>
1B	0B	<p>Software Auto Reload mode :</p> <p>WUT starts running when <b>PMSWCR3.WUTEN</b> = 1 and <b>WUTMODE</b> = 0 is set. <b>PMSWCR0.WUTWKEN</b> = 1 is set to activate system wake-up from standby state. <b>PMSWSTAT.WUTRUN</b> bit is set indicating that WUT timer is currently running.</p> <p><b>PMSWUTCNT.WUTCNT</b> bit field indicates the actual counter value. On counter underflow, WUT is automatically reloaded with WUTREL value. During Standby, WUT underflow triggers system wake-up if <b>PMSWSTAT2.WUTWKEN</b> is set and <b>PMSWSTAT2.WUTWKP</b> flag is set. During Run, Idle or Sleep modes, WUT underflow triggers an interrupt request and <b>PMSWSTAT2.WUTWKP</b> flag is set. WUTREL reload value shall not be updated in this state.</p> <p>On wake-up, the <b>PMSWSTAT2.WUTWKP</b> flag shall be cleared by <b>PMSWSTATCLR.WUTWKPCLR</b> bit. In case of un-serviced consecutive counter underflow events, <b>PMSWSTAT2.WUTOVRUN</b> flag is set to indicate an over-run wake-up event.</p> <p>Interrupt over-run event can be detected via SRC.IOV bit during RUN mode.</p>
1B	1B	<p>Standby Auto Stop mode:</p> <p>The mode is selected by setting <b>PMSWCR3.WUTEN</b> = 1, <b>WUTMODE</b> = 1 is set. <b>PMSWCR0.WUTWKEN</b> = 1 is set to activate system wake-up from standby state.</p> <p>WUT starts running only when Standby mode is entered. On counter underflow, WUT stops running and the wake-up of system is triggered and <b>PMSWSTAT2.WUTWKP</b> flag is set.</p> <p>WUT starts running again on the next Standby mode entry. The intention is to have the timer running only during the Standby state.</p> <p><b>PMSWUTCNT.WUTCNT</b> reloads <b>PMSWCR3.WUTREL</b> value on a wake-up.</p> <p><b>PMSWSTAT.WUTRUN</b> reads 0 after a wake-up.</p> <p><b>PMSWCR3.WUTREL</b> reload value shall not be updated in this state.</p> <p>On wake-up, the <b>PMSWSTAT2.WUTWKP</b> flag shall be cleared by <b>PMSWSTATCLR.WUTWKPCLR</b> bit. In case system was woken up by other wake-up triggers while WUT was still running, an interrupt request is generated on WUT underflow. In case of un-serviced consecutive counter underflow events, <b>PMSWSTAT2.WUTOVRUN</b> flag is set to indicate an over-run wake-up event.</p> <p>Interrupt over-run event can be detected via SRC.IOV bit during RUN mode.</p>

In case of timer overflow, an interrupt is issued on the interrupt node SRC\_PMSx (WUT). Wake-up Timer reload value maybe trimmed during Run mode by comparing the time stamp on a WUT underflow captured by a GTM-TIM or CCU6x based on a more precise clock as shown in [Figure 1.18](#). This allows to compensate on short term the 70 kHz (fSB) clock source variations owing to technology, voltage and temperature.

## Power Management System (PMS)

### 11.2.3.4.8 Entering Standby Mode (only VEVRSB domain supplied)

The Standby Mode entry may be requested via VEXT supply ramp-down triggered by a secondary SWDUV under-voltage event if configured in **PMSWCR0.VEXTSTBYEN** bits.

The Standby Mode entry may be requested by writing to PMCSR<sub>x</sub> register to set bit field REQSLP = 11<sub>B</sub> or via ESR1 (NMI) assertion as configured in SCU\_PMSWCR1.STBYEV bits.

Standby mode via SW may be entered based on a singular decision from a master CPU based on the configuration in the CPUSEL register. It may also be entered as a unanimous decision of all the CPUs when ALL of the PMCSR<sub>x</sub> registers in the system request it and PMSWCR1.CPUSEL = 111<sub>B</sub>. Each PMCSR<sub>x</sub> register is written by the corresponding CPU<sub>x</sub>.

Before entering Standby mode, modules may be sequentially shut off to avoid large load jumps.

- All peripherals and interrupts associated with CPUs except master CPU are switched off. This is to avoid wake-up of the CPUs once they are put into IDLE state.
- All CPUs except the master CPU are sequentially put into IDLE state. All watchdogs may be disabled or re-configured for slower modes. Peripherals module clocks are switched off in the respective CLC.DISR registers.
- Master CPU frequency reduction in steps compliant to load jump constraints. Master CPU code execution switched from Flash to PSPR RAM. Flash modules may be deactivated.
- System Clock is switched to internal 100 MHz clock source. System PLL & Peripheral PLL are switched off. Clock dividers are programmed to lower values.
- Standby SMU module shall be disabled via CMD\_STDBY.SMUEEN before going into Standby mode.
- SCU\_PMSWCR1.IRADIS bit set to disable Idle Request Acknowledge sequence activation for fast Standby Mode entry. System and Application reset generation possibilities are disabled. Only remaining reset possibility in this phase is via PORST or power-fail
- Standby RAM block selected via **PMSWCR0**.STBYRAMSEL bits. Dcache write back to be executed before Standby entry.
- Select the 70 kHz Standby clock source ( fSB ) via **PMSWCR4**.SCRCLKSEL bits. Configure the blanking filter appropriately via **PMSWCR0**.BLNKFIL bits.
- Configure pad state via **PMSWCR5**.TRISTREQ bit. All pads may be set into tristate or have pull-up device active. Regardless of the **PMSWCR5**.TRISTREQ setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate after standby mode entry. Configure **PMSWCR5**.ESR0TRIST bit to set ESR0 behavior as reset output active or tristate. In case of HWCFG [2:1,6] pins it is recommended to tie them to external pull devices.
- SCR may be kept running before entering the Standby state. The shared ports supplied by VEVRSB are configured either to be in tristate during standby or left to the control of SCR. The enabling of wake-up from SCR via **PMSWCR0**.SCRWKEN should be programmed when SCR is running at 20 MHz.
- Wake-up Timer may also be kept running before entering the Standby state. The wake-up from WUT may be activated via **PMSWCR0**.WUTWKEN bit.
- Wake-up via PORST, Pin A, ESR0 & ESR1 pins is not supported during Standby mode as VEXT will be ramped down on Standby entry. Therefore the respective **PMSWCR0**.PORSTWKEN, PINAWKEN, ESR0WKEN and ESR1WKEN wake-up configuration bits shall be disabled to avoid spurious wake-up triggers. It should be taken care that SCR is not reset on a standby entry by clearing **PMSWCR4**.PORSTREQ bit field.
- Enable wake-up on VEXT supply ramp-up via **PMSWCR0**.PWRWKEN bit. It need to be ensured that both PWRWKEN and VEXTSTBYEN register bits are both set before entering Standby mode. VEXTSTBYEN register bitfield shall be set to ensure that when VEXT supply is removed during Standby state, no LVD reset is generated consequently exiting from Standby mode.
- In case standby entry is triggered by VEXT supply ramp down, the threshold is configured in **EVRUVMON**.SWDUVVAL and transition condition in **EVRMONCTRL** register respectively. In case of nominal

## Power Management System (PMS)

VEXT supply voltage of 5 V, it is recommended to configure SWDUVVAL register bitfield at 4 V for standby entry to have adequate distance to primary reset levels as well as operational region limits. In case of nominal VEXT supply voltage of 3.3V, it is recommended to configure SWDUVVAL register bitfield at 3.1 V for standby entry above primary reset levels. Nevertheless since there is only a minimal margin to reset levels in this case, Standby entry is additionally triggered by the crossing of primary undervoltage limits if VEXTSTBYEN register bit is set to ensure Standby entry in case of fast VEXT slopes. The parasitic diode path from VDDP3 to VEXT will keep the VEXT voltage at (VDDP3 - diode drop), so it should be ensured that SWDUVVAL is configured above (VDDP3 - diode drop) for standby entry if VDDP3 and VEXT supply rails are separately supplied. The selection of only VEXT supply voltage monitoring in **EVRMONCTRL** would reduce the secondary monitor standby entry latency time to (tMON/3). Configure Standby entry event in **PMSWCRO.VEXTSTBYEN** register bit.

- It shall be ensured that the primary under-voltage reset monitors are active before Standby entry is triggered and shall not be disabled in **EVRRSTCON** register.
- The external regulator is communicated to switch off VEXT supply. A controlled ramp-down of VEXT slope during Standby entry is recommended from external regulator (E.g - 0.5V/ms to 1.5V/ms). It need to be ensured that the VEXT supply is ramped below VEXT LVD reset level after standby entry before blanking filter time has expired. This is to avoid an immediate wake-up triggered by the residual VEXT voltage if it is above VEXT LVD reset level after blanking time has expired. It need to be also ensured that VDD and VDDP3 supply rails are consequently switched off after VEXT ramp down to reduce standby current within blanking filter time.
- All xWKP / xOVRUN flags activated by respective xWKEN bits shall be cleared before renewed Standby entry request, otherwise System will remain in Operation state and not enter Standby state. Standby request is issued via VEXT supply undervoltage event or via SW or NMI event. Once Standby entry event is recognised, the primary under-voltage reset generation is disabled and Standby RAM supply is switched from VDD to VDDP3 within a single 25 MHz clock cycle. During Standby state entry, the wake-up logic is unable to detect wake-up events for a minimal time period less than 300 ns, therefore it need to be ensured that the wake-up pulse is asserted long enough that wake-up is detected. On entry into Standby mode, blanking filter is activated. The external standby regulator continues to supply the Standby domain via VEVRSB supply pin. Blanking filter is always activated on entry to Enter Standby state.
- When entering standby with external pass devices or external MOSFET complementary switch in case of EVRC regulator, it should be taken care that when VEXT supply is ramped down also the supply to the pass devices/MOSFET is also ramped down along with VEXT supply. Otherwise it may happen, that the VEXT supply is still held high via the diode path through the VGATE pins to VEXT supply rail. This would lead to immediate wake-up after blanking time has expired as VEXT supply is above the wake-up voltage threshold between 2,6 - 2,97 V. Adequate blanking filter time shall be configured before entering standby to ensure that VEXT has ramped down completely within this time to avoid immediate wake-up.
- Standby request issued via REQSLP bit field or ESR1/NMI event.
- Select required edge configuration in SCU\_ESRCFG1.EDCON if ESR1 is used as a trigger for standby entry.

## Power Management System (PMS)

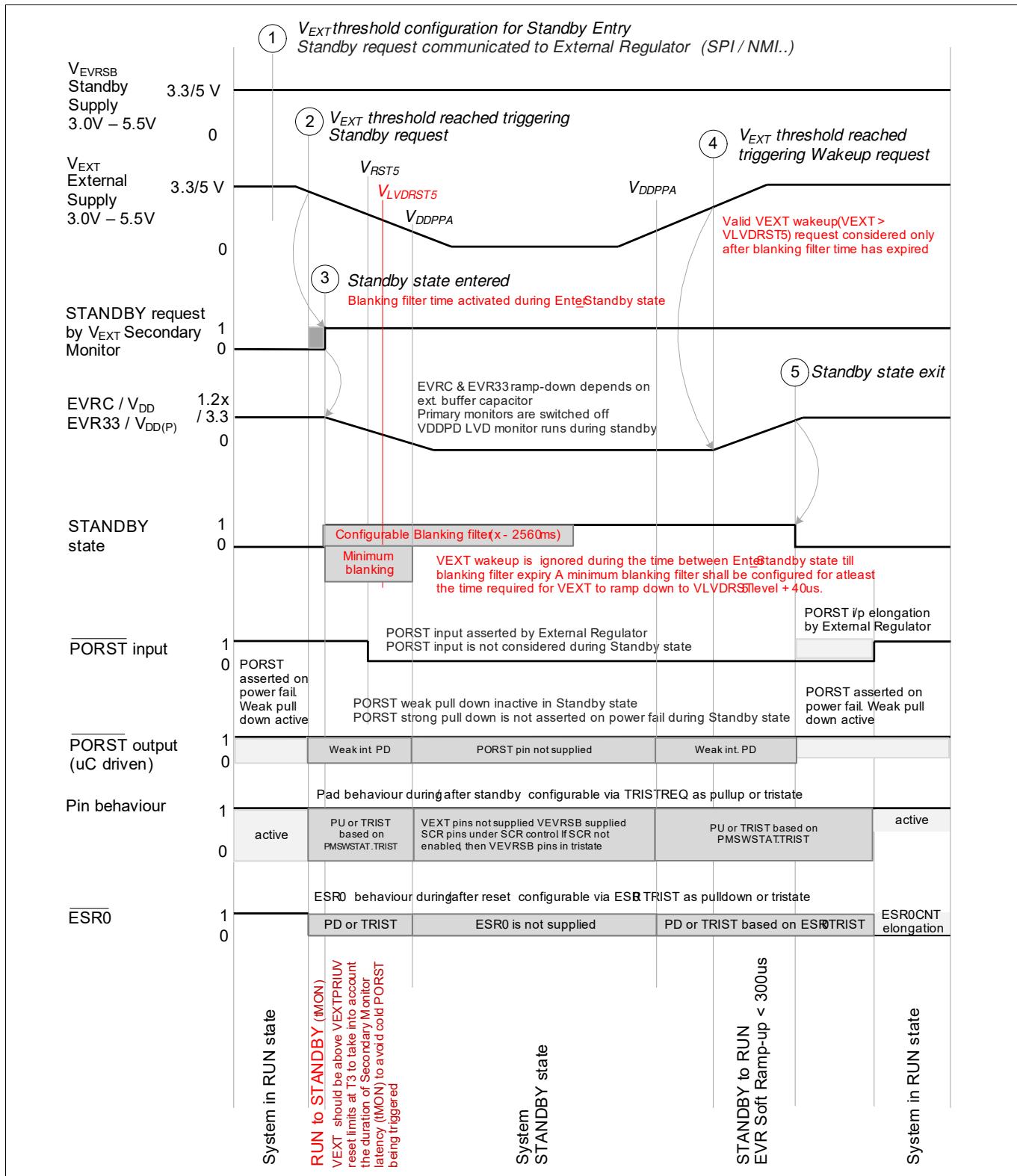


Figure 119 Standby entry on VEXT ramp-down and wake-up on VEXT ramp-up

## Power Management System (PMS)

### 11.2.3.4.9 Entering Standby Mode (both VEVRSB and VEXT domain supplied)

The Standby Mode entry may be requested by writing to PMCSR<sub>x</sub> register to set bit field REQSLP = 11<sub>B</sub> or via ESR1 (NMI) assertion as configured in SCU\_PMSWCR1.STBYEV bits.

Standby mode via SW may be entered based on a singular decision from a master CPU based on the configuration in the CPUSEL register. It may also be entered as a unanimous decision of all the CPUs when ALL of the PMCSR<sub>x</sub> registers in the system request it and PMSWCR1.CPUSEL = 111<sub>B</sub>. Each PMCSR<sub>x</sub> register is written by the corresponding CPUx.

Before entering standby mode, various modules should be sequentially shut off in a sequence mainly to avoid large current jump on standby entry.

- All peripherals and interrupts associated with CPUs except master CPU are switched off. This is to avoid wake-up of the CPUs once they are put into IDLE state.
- All CPUs except the master CPU are sequentially put into IDLE state. CPU watchdogs may be disabled or re-configured for slower modes. Peripheral module clocks are switched off in respective CLC.DISR registers.
- Master CPU frequency reduction in steps compliant to load jump constraints. Master CPU code execution is switched from Flash to PSPR RAM. Flash modules are deactivated.
- System Clock is switched to the internal 100 MHz clock source. System PLL & Peripheral PLL are switched off. Clock dividers are programmed to lower values.
- Standby SMU module shall be disabled via CMD\_STDBY.SMUEN before going into Standby mode.
- Set SCU\_PMSWCR1.IRADIS bit to disable Idle Request Acknowledge sequence activation for fast Standby Mode entry. This ensures that standby request is not blocked by a pending reset request / sequence.
- Select the Standby RAM block via **PMSWCR0**.STBYRAMSEL bits. Dcache write back to be executed before Standby entry.
- Select the 70 kHz Standby clock source ( fSB ) via **PMSWCR4**.SCRCLKSEL bits. Configure the blanking filter appropriately via **PMSWCR0**.BLNKFIL bits.
- Select the clock source which need to be active on entry into Standby Mode via **PMSWCR4**.SCRCLKSEL bits. Wake-up trigger edge configuration and filter activation is configured via **PMSWCR0**.xxxEDCON and **PMSWCR0**.xxxDFEN bits.
- Configure pad state via **PMSWCR5**.TRISTREQ bit. All pads may either be in tristate or have pull-up devices active. Regardless of the **PMSWCR5**.TRISTREQ setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate after standby mode entry. Configure **PMSWCR5**.ESR0TRIST bit to configure ESR0 behavior as reset output or tristate during Standby and on wake-up. In case of HWCFG [2:1,6] pins it is recommended to tie them to external pull devices.
- Wake-up Timer may also be kept running before entering the Standby state. The wake-up from WUT may be activated via **PMSWCR0**.WUTWKEN bit.
- Enable ESR1 or PINx pins for wake-up via **PMSWCR0**.xxxWKEN bits.
- SCR may be kept running before entering the Standby state. The shared ports supplied by VEVRSB are configured either to be in tristate during standby or left to the control of SCR. The enabling of wake-up from SCR via **PMSWCR0**.SCRWKEN should be programmed when SCR is running at 20 MHz.
- All xWKP / xOVRUN flags activated by respective xWKEN bits shall be cleared before renewed Standby entry request, otherwise System will remain in Operation state and not enter Standby state. Configure Standby Entry event in SCU\_PMSWCR1.STBYEV register bits.
- It shall be ensured that the primary under-voltage reset monitors are active before Standby entry is triggered and shall not be disabled in **EVRRSTCON** register.
- Standby request issued via REQSLP bit field or ESR1/NMI event. An orderly shut down of various sub-systems is triggered to enter Standby mode. Once Standby entry request is recognised, the primary under-voltage

## Power Management System (PMS)

reset generation is disabled and Standby RAM supply is switched from VDD to VDDPD within a single 25 MHz clock cycle. During Standby state entry, the wake-up logic is unable to detect wake-up events for a minimal time period less than 300 ns, therefore it need to be ensured that the wake-up pulse is asserted long enough that wake-up is detected. Blanking filter is always activated on entry to Enter\_Standby state. It need to be ensured that VDD and VDDP3 supply rails are consequently switched off after entry to standby to reduce standby current.

- Select required edge configuration in SCU\_ESRCFG1.EDCON if ESR1 is used as a trigger for standby entry.

### 11.2.3.4.10 State during Standby Mode

The Standby RAM (DLMU RAM of CPU0 and CPU1), the 8 bit Standby controller, the shared ports and the wake-up logic are kept alive in Standby mode. PORST pin, ESRx pins and PIN A provides wake-up function if VEXT is supplied. In case of wake-up on VEXT supply ramp-up and only VEVRSB is supplied, cold PORST function is resumed only after all supplies have ramped up.

All other pins are set in their default reset state. The default pin behavior during standby and after wake-up may be configured as pull-up or tristate accordingly by TRISTREQ register bit. Regardless of the [PMSWCR5.TRISTREQ](#) setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate after standby mode entry. All pins can be set into tristate except the TESTMODE pin where the internal pull-up is active also during Standby mode. The ESR0 pin may be configured as reset output or tristate during Standby mode by configuring ESR0TRIST bit. The shared port supplied by VEVRSB may retain their state. It need to be ensured by the external regulator that the VEVRSB voltage is within the operational region during Standby state.

The SCR continues to operate as a stand-alone 8 bit controller executing the intended operations. It should be ensured that the shared ports are configured in the corresponding port registers and the ownership of the pins are assigned either to the SCR or the main domain. In case the SCR needs to drive outputs during Standby mode, the default clock may need to be switched from 70 kHz to 20 MHz clock source. The SCR may request the EVR to activate or deactivate the 20 MHz clock. Analog conversions maybe carried out using SCR ADCOMP unit. SCR may be programmed to issue wake-up based on inputs from internal modules or shared pins. When the wake-up of the main core domain is required, the SCR issues a wake-up request via SCRWK bit in STDBYWKP register as documented in the SCR SCU.

### 11.2.3.4.11 Exiting Standby Mode - Wake-up event

The wake-up trigger in case of Standby mode where VEVRSB domain is only supplied may happen

- On a VEXT Supply ramp up after the blanking filter time has expired. SCR may be active. Wake-up can indirectly be triggered via SCR by communicating to external regulator to request the ramp-up of VEXT voltage. The wake-up reason could be any SCR event, Pin B edge transition or WUT Wake-up. Pin B edge transition or WUT Wake-up is also communicated to SCR as shown in [Figure 120](#).

After VEXT wakeup is recognised, it is expected that the VEXT supply is stable afterwards. In case of immediate VEXT powerfail consequent to VEXT wake-up, LVD reset or cold PORST may be triggered.

It is expected that the VEXT Supply has ramped down within the configured blanking filter time. Blanking filter shall be configured for atleast the time required for VEXT to ramp down to VLDRST5 level + 40 us. VEXT wakeup is ignored during the time between Enter Standby state till blanking filter expiry. A wake-up is triggered when the VEXT Supply is above the wake-up threshold of VLDRST5 for a time duration greater than 20 us indicated by event 4 in [Figure 119](#). Wake-up triggered on a VEXT Supply ramp-up is indicated in [PMSWSTAT2.PWRWKP](#) register bit and shall be cleared by [PMSWSTATCLR.PWRWKP](#) clear register bit.

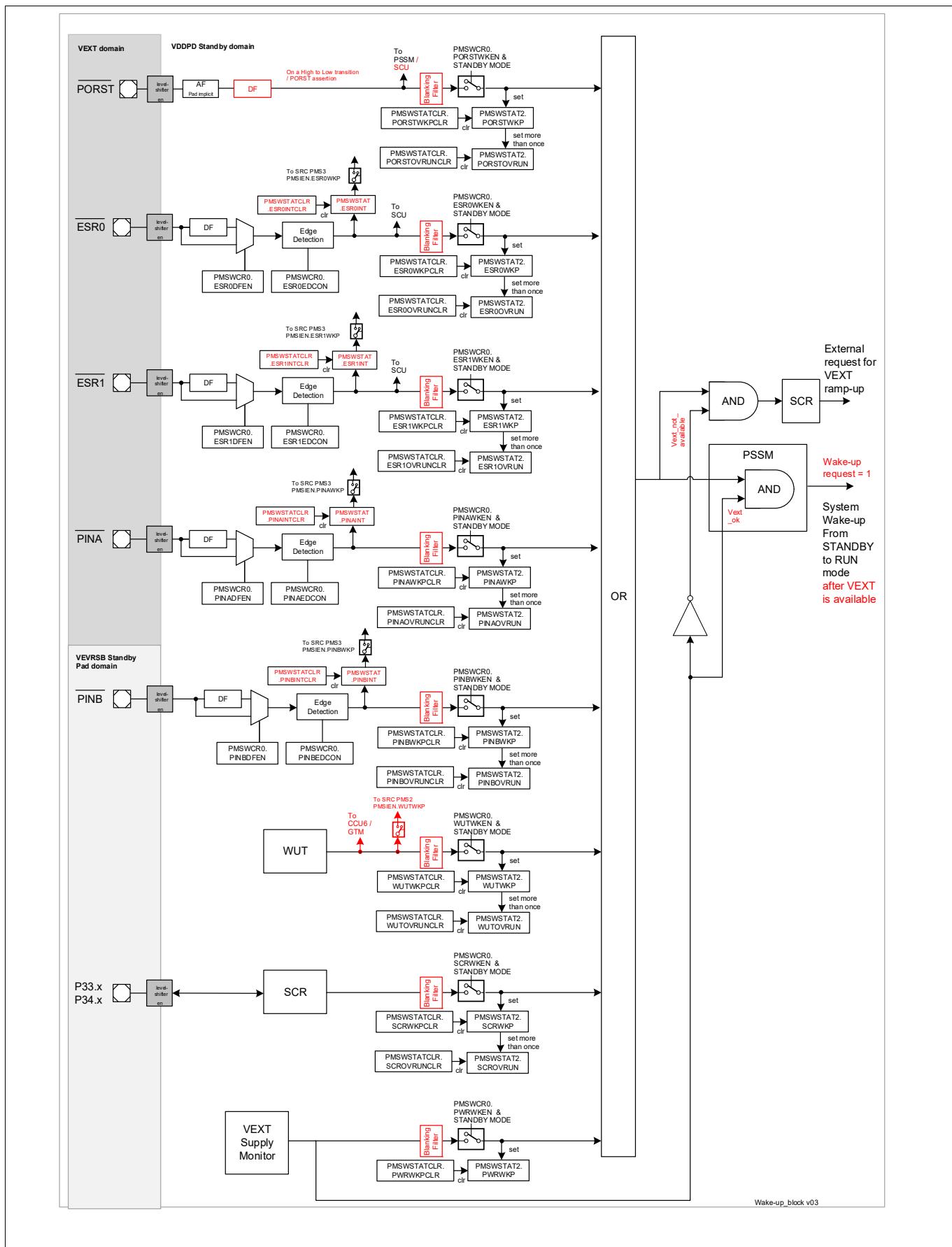
The wake-up event in case of Standby mode where both VEVRSB and VEXT domain supplied may happen after the blanking filter time has expired on following events. ESRx / PINx edge, WUT underflow or SCR wakeup is ignored during the time between Enter\_Standby state till blanking filter expiry. xWKP / xOVRUN flags are only set during Standby mode after Blanking Filter expiry when the respective wake-up event happens.

## Power Management System (PMS)

- ESR1 edge transition (NMI trap): **PMSWSTAT2.ESR1WKP** set on wake-up. **PMSWSTAT2.ESR1OVRUN** set to indicate overrun behavior in case of multiple un-serviced wake-ups.
- Pin A or Pin B edge transition (P14.1 or P33.12): **PMSWSTAT2.PINxWKP** set on wake-up. **PMSWSTAT2.PINxOVRUN** set to indicate overrun behavior in case of multiple un-serviced wake-ups.
- Wake-up from SCR via register STDBYWKP.SCRWKP in turn caused by following events. **PMSWSTAT2.SCRWKP** set on wake-up. **PMSWSTAT2.SCROVRUN** set to indicate overrun behavior in case of multiple un-serviced wake-ups.
  - Edge transitions at the shared ports
  - RTC interrupt
  - SCR watchdog overflow
  - Selected interrupts from communication modules
  - ADCOMP analog channel compare event.
- Wake-up from WUT

The main EVRC and EVR33 regulators are ramped up on wake-up based on the earlier latched configuration in **PMSWSTAT.HWCFCGEVR** register bits. On wake-up, all pads are either in tristate or are connected to pull-ups as indicated in **PMSWSTAT.TRIST** register bit. ESR0 behavior is indicated in **PMSWSTAT.ESR0TRIST** register bit. If Standby RAM was supplied during Standby state, it is indicated in **PMSWSTAT2.STBYRAM** register bits. Additional RAM integrity checks may be carried out after wake-up. RSTSTAT.STBYR bit indicates that the supply was reliable during Standby. The wake-up and over-run status flags are set in **PMSWSTAT2** register and shall be cleared by **PMSWSTATCLR** register. The wake-up time is nearly the same as the normal boot time as EVR need to be started and firmware need to be consequently executed.

## Power Management System (PMS)



**Figure 120 Wake-up Overview**

## Power Management System (PMS)

### 11.2.3.4.12 Exiting Standby Mode - Power Fail or Reset event

A power fail event of the Standby supply (VEVRSB pin) during Standby mode may inevitably result in the loss of Standby RAM contents. Consequently, LVD reset event is issued and the Standby domain is set into reset. EVR Pre-regulator under-voltage violation is indicated in RSTSTAT.STBYR flag which can be used as an indication whether Standby supply fail had happened and Standby RAM contents are reliable. Cold PORST flags RSTSTAT.SWD, EVRC and EVR33 would always be set after wake-up from STANDBY mode as these domains may be devoid of power during STANDBY to eliminate leakage current. It is recommended to keep a copy of the critical data also in Dflash in order to mitigate the effects if unwanted power fail events cannot be avoided.

In case of VEXT supply wake-up, PORST pin, ESRx pins and PIN A input are not evaluated during Standby mode as it is supplied by VEXT domain which is switched off during the STANDBY mode. In case VEXT domain is supplied during STANDBY mode, the Standby domain is woken up on PORST assertion depending on **PMSWCR0.PORSTWKEN** bit. **PMSWCR0.PORSTWKEN** is by default set to 1 to ensure wake-up on PORST assertion during STANDBY mode. The SCR may also set into reset simultaneously depending on **PMSWCR4.PORSTREQ** bit. The device boots up ramping up the regulators followed by firmware execution similar to a normal device start-up. On PORST wake-up, **PMSWSTAT2.PORSTWKP** event flag is set providing information as to the wake-up source. It should be taken care after wake-up to clear the event flags via **PMSWSTATCLR.PORSTWKPCCLR** register. In case new PORST wake-up events are captured while **PMSWSTAT2.PORSTWKP** flags are still set, then **PMSWSTAT2.PORSTOVRRUN** flags are set to indicate an overrun state owing to consecutive un-serviced wake-up events. The overrun flag is cleared via **PMSWSTATCLR.PORSTOVRUNCLR** bit.

The Standby RAM contents are kept intact after a wake-up caused by PORST assertion. In case of VDD supply under-voltage condition during wake-up phase, it is ensured that the Standby RAM is kept supplied by VDDPD until VDD is back in operational range to avoid Standby RAM data loss or corruption during transition. Reset is propagated to external devices via the ESR0 pin on exit from Standby mode depending on **PMSWCR5.ESR0TRIST** configuration. Firmware may elongate ESR0 reset output depending on Flash configuration.

Additional PORST digital filter activated via **PMSWCR5.PORSTDF** bit provides additional spike filtering of at least tPORSTDF duration to provide enhanced immunity against spurious spikes. This is in addition to the inherent analog PORST filter delay of the PORST pad / pin as documented in the datasheet. After cold PORST the additional PORST digital filter delay is by default inactive. If VEXT is supplied, PORST (high to low) during Standby state after blanking filter expiry triggers wake-up.

**Table 302 PORST pin assertion behavior on PMS and SCR subsystem during power modes**

Reaction to PORST pin assertion	RUN mode SLEEP mode	STANDBY mode
No reaction	No effect on PMS domain. No reaction on SCR if <b>PMSWCR4.PORSTREQ</b> = 0 but an SCR_NMI is triggered via the PMSWCR2.RST bit.	No effect on PMS domain if <b>PMSWCR0.PORSTWKEN</b> = 0. No reaction on SCR if <b>PMSWCR4.PORSTREQ</b> = 0
Wake-up	No effect on PMS or SCR domain as the system is already awake	PMS Standby to RUN transition takes place on PORST assertion when VEXT is supplied if <b>PMSWCR0.PORSTWKEN</b> =1(default)
Reset	No reset of PMS domain SCR is reset if <b>PMSWCR4.PORSTREQ</b> = 1 (default)	No reset of PMS domain SCR is reset if <b>PMSWCR4.PORSTREQ</b> = 1 (default)

## Power Management System (PMS)

### 11.2.3.5 Load Jump Sequencing and Voltage Droop

Load jumps lead to consequent voltage overshoots / undershoots which need to be limited within the regulator dynamic specification and operational bounds of the supply rail. The initial phase after the load jump is buffered by the external capacitor which consequently leads to a linear discharge of the capacitor. Consequently the regulator feedback loop recognizes the deviation in voltage and reacts to the jump by changing the control output. The dimensioning of the capacitor results mainly from the load jump amplitude, ESR of the capacitor, the permissible voltage deviation and reaction time of the regulator. The capacitor size in turn has a tangible impact on BOM cost and PCB space. Minimizing peak-to-peak voltage deviation in the face of such large dynamic changes in load current need to be actively managed if large amounts of output capacitance are to be avoided.

If  $V_{DD}$  supply is generated by the internal EVRC regulator, the voltage transients owing to load jumps on core  $V_{DD}$  supply rail need to be restricted within  $V_{DD\_SETPOINT} + 8\% - 6\%$ . This includes a static accuracy of  $V_{DD\_SETPOINT} \pm 2\%$  and consequently  $+ 6\% - 4\%$  remaining for dynamic regulation.

In case of external  $V_{DD}$  supply, the voltage transients owing to load jumps on core  $V_{DD}$  supply rail need to be restricted within  $V_{DD\_SETPOINT} \pm 5\%$ . This includes a static accuracy of  $V_{DD\_SETPOINT} \pm 2\%$  and consequently  $\pm 3\%$  remaining for dynamic regulation.

Load jumps may be triggered by software or user driven actions or asynchronous hardware events. During software triggered non reset events, it is recommended to limit the load jumps ( $dIEXT/dt$ ,  $dIDD/dt$ ) to a maximum of 100 mA with 100  $\mu$ s settling time. For example, during clock ramp-up phase it is recommended to limit the clock switching steps so as not to violate this limit.

#### Handling load jump hardware events triggered asynchronously - Resets and NMI

In case of typical application load jump events, triggered asynchronously, like Non Maskable Interrupts and reset events, namely application, system and warm power-on reset requests, measures are built in to ensure that voltage overshoots are kept within bounds by load sequencing mechanisms or by means of register configuration.

In case of an NMI event, during RUN mode or waking up from SLEEP mode, the device may activate a large number of hitherto dormant circuits and wake-up the CPUs simultaneously resulting in a large change in load current. To avoid a large load jump on an NMI event, it needs to be ensured that only one CPU is triggered by the NMI or woken out of SLEEP mode and that other CPUs are still in IDLE mode. The other CPUs are consequently started one after another with adequate delay in between during start-up phase. The active CPU woken up on an NMI request is selected based on TRAPDIS0 and TRAPDIS1 register configurations.

In case of an Application Reset, System Reset or warm Power-On Reset request, the port pins are immediately set into reset state. Consequently the CPUs are ramped down in a sequence during the first 80  $\mu$ s immediately after the warm reset request. Finally after 180  $\mu$ s after reset request, the asynchronous reset event is issued to the device allowing to limit the maximum warm reset load jump to roughly half of the total dynamic IDD (IDDRAIL minus IDDPORST) current. It needs to be ensured that the VEXT, VDDP3 and VDD supply voltages are above the minimum operational voltage limits during the total reset phase of 180  $\mu$ s after warm reset request not to trigger a cold power-fail reset. Larger overshoots are tolerable during and after reset phase for a certain cumulated time but must be limited to operational and absolute maximum voltage ratings as documented in the datasheet.

Load jump events caused by asynchronous failure events like PLL loss of lock or external oscillator watchdog event may lead to overshoots which cannot be sequenced owing to the inherent nature of failure.

#### Handling simultaneous load jump requests triggered by Software

Software triggered Load Jump events include ramping up / down of various system clock frequencies, activating additional CPUs, Power mode transitions, CPU throttling and idle requests, MTU Memory tests, LBIST tests and so forth.

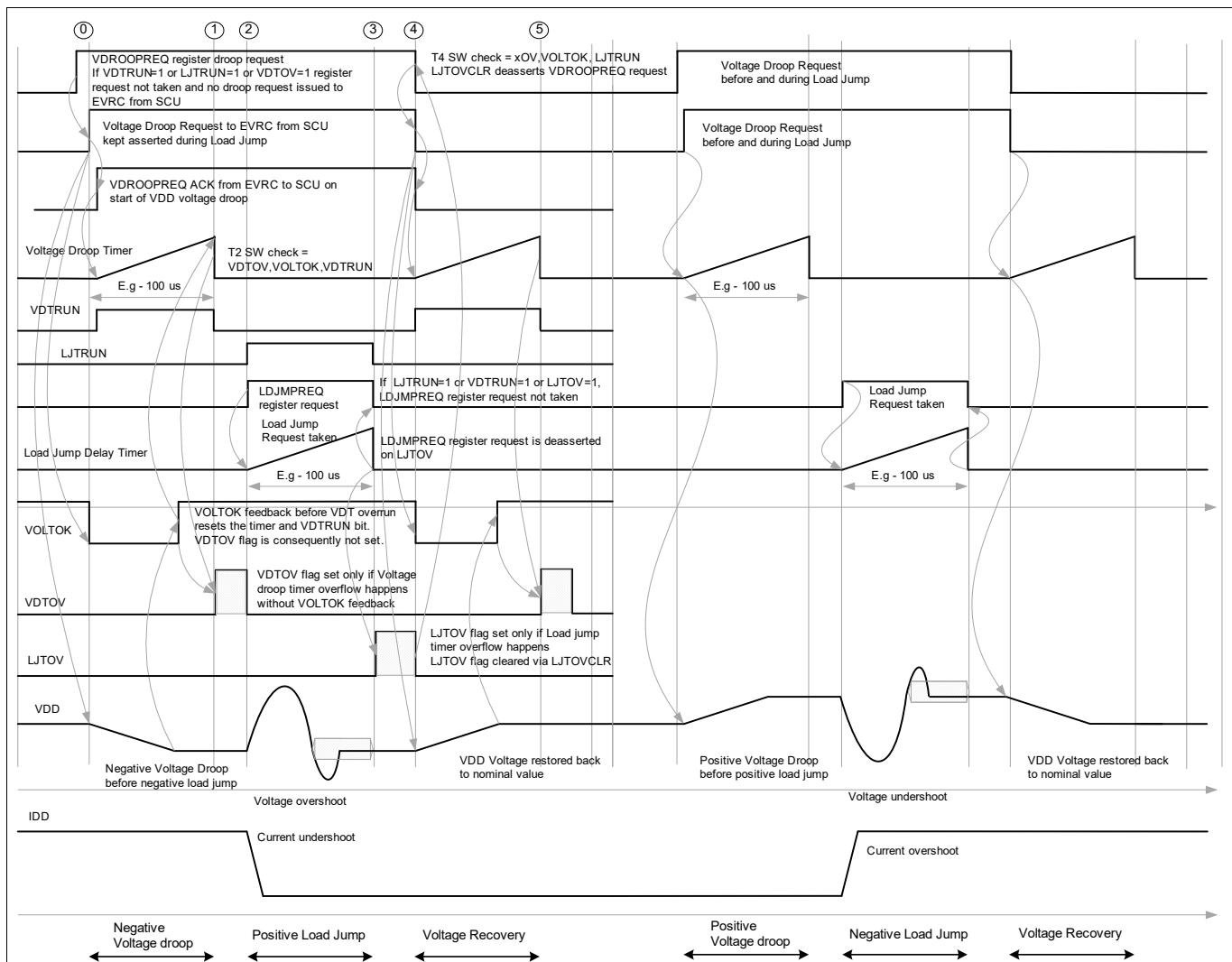
## Power Management System (PMS)

In case of software triggered events, it is possible to prepare by lowering or raising the voltage setpoint before the load jump is issued. A negative voltage droop may be done before a negative load jump and a positive voltage droop before a positive load jump respectively. Thus negative load jumps leading to voltage overshoots is compensated partly by the negative voltage droop and likewise positive load jumps leading to voltage undershoots is compensated partly by the positive voltage droop as shown in [Figure 121](#). The voltage droop is configured through SCU\_PMTCSR0.SDSTEP register bits. The voltage droop in positive or negative direction is issued via SCU\_PMTCSR3.VDROOPREQ register bits. In case a current Vdroop request is not active or the Voltage Droop Timer is not currently running indicated via SCU\_PMTCSR3.VDTRUN or the Load Jump Timer is not currently running indicated via SCU\_PMTCSR2.LJTRUN, a new Vdroop request is taken. Once a new voltage droop request is issued, **EVRSTAT**.SDVOK is reset and TC3xx need to wait for a certain time till the regulator has settled on the new value which is realized using a Voltage Droop Timer. Once the regulator has settled on the new value, **EVRSTAT**.SDVOK status bit is set again indicating the end of the Voltage Droop transition and SCU\_PMTCSR3.VDTRUN and SCU\_PMTCSR3.VDTCNT is reset by hardware. If SDVOK status bit is set by EVRC before compare match of VDT has occurred, VDTOV overflow bit is not set and overrun interrupt is not generated. The Voltage Droop Timer compare value is configured in SCU\_PMTCSR1.VDTCV register bits and the current value is indicated in SCU\_PMTCSR3.VDTCNT register bits. In case of a compare match, the overflow SCU\_PMTCSR3.VDTOV bit is set if enabled via SCU\_PMTCSR0.VDTOVEN register bits. In this case, overflow bit has to be explicitly cleared via SCU\_PMTCSR3.VDTOVCLR before a new request can be taken to support a sequential polling based approach. Furthermore, interrupt maybe activated on an overflow if SCU\_PMTCSR0.VDTOVIEN is enabled.

Simultaneous software triggered load jump events can be likewise avoided by checking whether a Load Jump is ongoing or the Load Jump Timer is currently running. The Load Jump Request is issued by triggering a compare and swap operation on SCU\_PMTCSR2 register. A CPU will access data from SCU\_PMTCSR2 register and will compare the current value with an expected value. The expected value is that there is no load jump currently ongoing and VDTRUN bit state is 0. If there is a match, the CPU will make the swap by setting SCU\_PMTCSR2.LDJMPREQ variable and starting the timer. Obviously if multiple CPUs are making this operation simultaneously only one CPU will succeed and others will fail the compare and swap operation when the Load Jump Timer would be running. The idea is to prevent multiple CPUs from doing load jumps simultaneously by treating Load Jump as a critical section and ensuring that only a single CPU can check and issue a load jump request atomically. Once a load jump request is taken, a timer is started and other CPUs have to wait till the regulator output has been restored to the setpoint value and ensures adequate regulator reaction time. The other CPUs are not blocked during the waiting period instead they can continue with some other operations or try to make the request again at a later point of time.

Thus negative load jumps and positive load jumps are followed by a blanking period using a Load Jump Timer as shown in [Figure 121](#). The Load Jump Timer is configured through SCU\_PMTCSR0 register. The load jump request is issued via SCU\_PMTCSR2.LDJMPREQ register bits. If a current Load Jump request is not active or the Load Jump Timer is not currently running indicated via SCU\_PMTCSR2.LJTRUN or the Voltage Droop Timer is not currently running indicated via SCU\_PMTCSR3.VDTRUN, a new Load Jump request is taken. Once a new Load Jump request is issued, the device needs to wait for a certain time till the regulator has reacted to the jump which is realized using a Load Jump Timer. The Load Jump Timer compare value is configured in SCU\_PMTCSR1.LJTCV register bits and the current value is indicated in SCU\_PMTCSR2.LJTCNT register bits. In case of a compare match, the overflow SCU\_PMTCSR2.LJTOV bit is set if enabled via SCU\_PMTCSR0.LJTOVEN register bits. In this case, overflow bit has to be explicitly cleared via SCU\_PMTCSR2.LJTOVCLR before a new request can be taken to support a sequential polling based approach. Furthermore, an interrupt maybe activated on an overflow if SCU\_PMTCSR0.LJTOVIEN is enabled. Overflow bit is routed to an OS interrupt to schedule the next current jump. Overflow bit maybe masked or used in the compare and swap operation if SCU\_PMTCSR0.LJTOVEN is set to ensure that explicit clear of the time out has happened before issuing a new request. SCU\_PMTCSR2.LJTOVCLR also clears SCU\_PMTCSR3.VDROOPREQ and SCU\_PMTCSR2.LDJMPREQ request.

## Power Management System (PMS)



**Figure 121 Load jumps and Voltage Droop**

---

**Power Management System (PMS)**

### 11.2.3.6 Core Die Temperature Sensor (DTSC)

The Core Die Temperature Sensor (DTSC) generates a measurement result that indicates directly the current temperature. The DTSC measures the temperature with an accuracy within ( $T_{NL} + T_{CALACC}$ ) parameter limits within the TSR temperature range documented in the datasheet. The result of the measurement is updated periodically in DTSCSTAT.RESULT register bit field with a resolution less than 1/5th of a degree Kelvin. The Die Temperature Sensor is available after an application reset release on a device start-up and temperature measurements are carried out continuously during normal RUN / SLEEP modes once DTSC is enabled. The Die Temperature Sensor and DTSLIM and DTSCSTAT registers are reset on an application reset.

The DTSC is enabled via DTSLIM.DTSEN register bitfield. The DTS start-up is completed after a nominal 20us delay after DTSLIM.DTSEN is set. After an ongoing temperature measurement is completed, DTSCSTAT.RESULT bit field is updated coherently with the new value. An interrupt service request (SRC\_SCUERU3) can be generated after a measurement is completed. DTS bandgap status is reflected in DTSLIM.BGPOK status flag. The DTS accuracy and measurement time is defined in the Data Sheet. The DTSLIM register shall be updated before enabling DTSC via DTSLIM.DTSEN register bitfield.

Die temperature upper and lower limits are configured in DTSLIM.UPPER and LOWER register bits. On violation of these limits, DTSLIM.UOF and LLU status bits are set and alarms are forwarded to core SMU. After start-up or application reset, the DTSC limits have to be re-configured appropriately depending on the application before alarm reactions from SMU are activated. Only when a new DTSC conversion result is available, the DTSC comparators are consequently triggered to check the actual DTSCSTAT.RESULT against the upper and lower limits.

DTSCCON and DTSCBGOCTRL register can be also changed after DTSC is enabled on the fly for test purposes.

## 11.3 Registers

## Power Management System (PMS)

### 11.3.1 Power Management Control Registers (PMS)

**Table 303 Register Address Space - PMS**

Module	Base Address	End Address	Note
(PMS)	F0240000 <sub>H</sub>	F0241FFF <sub>H</sub>	
PMS	F0248000 <sub>H</sub>	F02481FF <sub>H</sub>	FPI slave interface

**Table 304 Register Overview - PMS (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
ID	Identification Register	0008 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">83</a>
EVRSTAT	EVR Status Register	002C <sub>H</sub>	U,SV	BE	See page <a href="#">83</a>	<a href="#">83</a>
EVRADCSTAT	EVR Primary ADC Status Register	0034 <sub>H</sub>	U,SV	BE	LVD Reset	<a href="#">88</a>
EVRRSTCON	EVR Reset Control Register	003C <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">90</a>	<a href="#">90</a>
EVRRSTSTAT	EVR Reset Status Register	0044 <sub>H</sub>	U,SV	BE	See page <a href="#">93</a>	<a href="#">93</a>
EVRTRIM	EVR Trim Control Register	004C <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">94</a>	<a href="#">94</a>
EVRTRIMSTAT	EVR Trim Status Register	0050 <sub>H</sub>	U,SV	BE	See page <a href="#">96</a>	<a href="#">96</a>
EVRMONSTAT1	EVR Secondary ADC Status Register 1	0060 <sub>H</sub>	U,SV	BE	See page <a href="#">97</a>	<a href="#">97</a>
EVRMONSTAT2	EVR Secondary ADC Status Register 2	0064 <sub>H</sub>	U,SV	BE	See page <a href="#">98</a>	<a href="#">98</a>
EVRMONCTRL	EVR Secondary Monitor Control Register	0068 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">99</a>	<a href="#">99</a>
EVRMONFILT	EVR Secondary Monitor Filter Register	0070 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">105</a>	<a href="#">105</a>
PMSIEN	PMS Interrupt Enable Register	0074 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">107</a>	<a href="#">107</a>
EVRUVMON	EVR Secondary Under-voltage Monitor Register	0078 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">110</a>	<a href="#">110</a>
EVROVMON	EVR Secondary Over-voltage Monitor Register	007C <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">111</a>	<a href="#">111</a>
EVRUVMON2	EVR Secondary Under-voltage Monitor Register 2	0080 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">112</a>	<a href="#">112</a>
EVROVMON2	EVR Secondary Over-voltage Monitor Register 2	0084 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">114</a>	<a href="#">114</a>
HSMUVMON	EVR Primary HSM Under-voltage Monitor Register	0088 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">115</a>	<a href="#">115</a>
HSMOVMON	EVR Primary HSM Over-voltage Monitor Register	008C <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">117</a>	<a href="#">117</a>
EVR33CON	EVR33 Control Register	0090 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">119</a>	<a href="#">119</a>

## Power Management System (PMS)

**Table 304 Register Overview - PMS (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
EVROSCCTRL	EVR Oscillator Control Register	00A0 <sub>H</sub>	U,SV	SV,SE,P	See page 119	119
PMSWCR0	Standby and Wake-up Control Register 0	00B4 <sub>H</sub>	U,SV	SV,SE,P	LVD Reset	154
PMSWCR2	Standby and Wake-up Control Register 2	00B8 <sub>H</sub>	U,SV	SV,SE,P	LVD Reset	158
PMSWCR3	Standby and Wake-up Control Register 3	00C0 <sub>H</sub>	U,SV	SV,SE,P	LVD Reset	159
PMSWCR4	Standby and Wake-up Control Register 4	00C4 <sub>H</sub>	U,SV	SV,SE,P	See page 161	161
PMSWCR5	Standby and Wake-up Control Register 5	00C8 <sub>H</sub>	U,SV	SV,SE,P	LVD Reset	163
PMSWSTAT	Standby and Wake-up Status Register	00D4 <sub>H</sub>	U,SV	BE	LVD Reset	164
PMSWSTAT2	Standby and Wake-up Status Register 2	00D8 <sub>H</sub>	U,SV	BE	LVD Reset	167
PMSWUTCNT	Standby WUT Counter Register	00DC <sub>H</sub>	U,SV	BE	LVD Reset	160
PMSWSTATCLR	Standby and Wake-up Status Clear Register	00E8 <sub>H</sub>	U,SV	SV,SE,P	LVD Reset	173
EVRSDDSTAT0	EVR SD Status Register 0	00FC <sub>H</sub>	U,SV	BE	See page 120	120
EVRSDDCTRL0	EVRC SD Control Register 0	0108 <sub>H</sub>	U,SV	SV,SE,P	See page 121	121
EVRSDDCTRL1	EVRC SD Control Register 1	010C <sub>H</sub>	U,SV	SV,SE,P	See page 123	123
EVRSDDCTRL2	EVRC SD Control Register 2	0110 <sub>H</sub>	U,SV	SV,SE,P	See page 128	128
EVRSDDCTRL3	EVRC SD Control Register 3	0114 <sub>H</sub>	U,SV	SV,SE,P	See page 130	130
EVRSDDCTRL4	EVRC SD Control Register 4	0118 <sub>H</sub>	U,SV	SV,SE,P	See page 134	134
EVRSDDCTRL5	EVRC SD Control Register 5	011C <sub>H</sub>	U,SV	SV,SE,P	See page 135	135
EVRSDDCTRL6	EVRC SD Control Register 6	0120 <sub>H</sub>	U,SV	SV,SE,P	See page 136	136
EVRSDDCTRL7	EVRC SD Control Register 7	0124 <sub>H</sub>	U,SV	SV,SE,P	See page 140	140
EVRSDDCTRL8	EVRC SD Control Register 8	0128 <sub>H</sub>	U,SV	SV,SE,P	See page 146	146
EVRSDDCTRL9	EVRC SD Control Register 9	012C <sub>H</sub>	U,SV	SV,SE,P	See page 147	147
EVRSDDCTRL10	EVRC SD Control Register 10	0130 <sub>H</sub>	U,SV	SV,SE,P	See page 148	148
EVRSDDCTRL11	EVRC SD Control Register 11	0134 <sub>H</sub>	U,SV	SV,SE,P	See page 149	149
EVRSDDCOEFF0	EVRC SD Coefficient Register 0	0148 <sub>H</sub>	U,SV	SV,SE,P	See page 124	124
EVRSDDCOEFF1	EVRC SD Coefficient Register 1	014C <sub>H</sub>	U,SV	SV,SE,P	See page 126	126
EVRSDDCOEFF2	EVRC SD Coefficient Register 2	0150 <sub>H</sub>	U,SV	SV,SE,P	See page 131	131

## Power Management System (PMS)

**Table 304 Register Overview - PMS (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
EVRSDCOEFF3	EVRC SD Coefficient Register 3	0154 <sub>H</sub>	U,SV	SV,SE,P	See page 132	132
EVRSDCOEFF4	EVRC SD Coefficient Register 4	0158 <sub>H</sub>	U,SV	SV,SE,P	See page 137	137
EVRSDCOEFF5	EVRC SD Coefficient Register 5	015C <sub>H</sub>	U,SV	SV,SE,P	See page 139	139
EVRSDCOEFF6	EVRC SD Coefficient Register 6	0160 <sub>H</sub>	U,SV	SV,SE,P	See page 142	142
EVRSDCOEFF7	EVRC SD Coefficient Register 7	0164 <sub>H</sub>	U,SV	SV,SE,P	See page 143	143
EVRSDCOEFF8	EVRC SD Coefficient Register 8	0168 <sub>H</sub>	U,SV	SV,SE,P	See page 144	144
EVRSDCOEFF9	EVRC SD Coefficient Register 9	016C <sub>H</sub>	U,SV	SV,SE,P	See page 145	145
AG2i_STDBY		0188 <sub>H</sub> +i*4	U,SV	SV,SE,P	LVD Reset	178
MONBISTSTAT	SMU_stdby BIST Status Register	0190 <sub>H</sub>	U,SV	BE	See page 178	178
MONBISTCTRL	SMU_stdby BIST Control Register	0198 <sub>H</sub>	U,SV	SV,SE,P	See page 178	178
CMD_STDBY	SMU_stdby Command Register	019C <sub>H</sub>	U,SV	SV,SE,P	See page 178	178
AG2iFSP_STDBY	SMU_stdby FSP Configuration Register	01A4 <sub>H</sub> +i*4	U,SV	SV,SE,P	See page 178	178
DTSSTAT	Die Temperature Sensor Status Register	01C0 <sub>H</sub>	U,SV	BE	See page 151	151
DTSLIM	Die Temperature Sensor Limit Register	01C8 <sub>H</sub>	U,SV	U,SV,P	See page 152	152
OTSS	OCDS Trigger Set Select Register	01E0 <sub>H</sub>	U,SV	U,SV,P	See page 175	175
OTSC0	OCDS Trigger Set Control 0 Register	01E4 <sub>H</sub>	U,SV	U,SV,P	See page 175	175
OTSC1	OCDS Trigger Set Control 1 Register	01E8 <sub>H</sub>	U,SV	U,SV,P	See page 177	177
ACCEN1	Access Enable Register 1	01F8 <sub>H</sub>	U,SV	SV,SE,32	Application Reset	179
ACCENO	Access Enable Register 0	01FC <sub>H</sub>	U,SV	SV,SE,32	Application Reset	178

---

## Power Management System (PMS)

### 11.3.1.1 Safety Flip-Flops

Safety flip-flops are special flip-flops that implement a hardware mechanism capable to detect single event effects that may lead to single event upsets (bit flip). The configuration and control registers that are implemented with safety flip-flops are:

- **EVRRSTCON**
- **EVRTRIM**
- **EVRMONCTRL**
- **EVRMONFILT**
- **EVRUVMON**
- **EVROVMON**
- **EVRUVMON2**
- **EVROVMON2**
- **HSMUVMON**
- **HSMOVMON**
- **EVROSCCTRL**
- **EVRSDCTRL0**
- **EVRSDCTRL1**
- **EVRSDCTRL2**
- **EVRSDCTRL3**
- **EVRSDCTRL4**
- **EVRSDCTRL5**
- **EVRSDCTRL6**
- **EVRSDCTRL7**
- **EVRSDCTRL8**
- **EVRSDCTRL9**
- **EVRSDCTRL10**
- **EVRSDCTRL11**
- **EVRSDCOEFF0**
- **EVRSDCOEFF1**
- **EVRSDCOEFF2**
- **EVRSDCOEFF3**
- **EVRSDCOEFF4**
- **EVRSDCOEFF5**
- **EVRSDCOEFF6**
- **EVRSDCOEFF7**
- **EVRSDCOEFF8**
- **EVRSDCOEFF9**
- **PMSWCR0**
- **PMSWCR5**

## Power Management System (PMS)

### 11.3.1.2 Power Supply Generation and Monitoring Control Registers

This section describes the kernel registers of the PMS module. Most of PMS kernel register names described in this section will be referenced in other parts of the Target Specification by the module name prefix “PMS\_”. All PMS registers are placed in the VDDPD Pre-Regulator domain. After a cold PORST, these registers may return the default isolation value or the updated value by the Firmware. Otherwise, a read will provide the value of the most recent write operation. In PMS subsystem some registers are reset with cold PORST which encompasses predominantly registers with EVR power generation and primary and secondary monitoring functions. This ensures that certain registers are not erroneously updated and the system does not end up in permanent reset situation. The registers covering standby and infrastructure functions however are reset with the EVR LVD (Low Voltage Detector Reset) master reset.

Tricore atomic instructions (LDMST, ST.T, SWAP.W, SWAPMASK.W, CMPSWAP.W) only write back bits that are changing their level. This leads to the fact that bits that are already set cannot be written with a 1 when using RMW instructions. No problem exists when using direct write instructions (e.g. ST.W). This affects the status bits in register DTSIM.LLU and UOF bits which are cleared by writing 1s.

#### Identification Register

<b>ID</b>															
<b>Identification Register</b>															
<b>(0008<sub>H</sub>) Application Reset Value: 00E8 C001<sub>H</sub></b>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>MODNUMBER</b>															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>MODTYPE</b>								<b>MODREV</b>							
r								r							

Field	Bits	Type	Description
<b>MODREV</b>	7:0	r	<b>Module Revision Number</b> This bit field indicates the revision number of the PMS module.
<b>MODTYPE</b>	15:8	r	<b>Module Type</b> This bit field is fixed coded as C0 <sub>H</sub> . It defines a 32-bit module.
<b>MODNUMBER</b>	31:16	r	<b>Module Number</b> This bit field defines the module identification number. The identification number for the PMS is 00E8 <sub>H</sub> .

#### EVR Status Register

The status registers EVRSTAT, EVRADCSTAT, EVRMONSTAT1, EVRMONSTAT2 and EVRSDSTAT0 are updated during Start-up and after every EVRx closed loop cycle with the actual status and therefore the read value may differ from the reset value. The over-voltage and under-voltage event flag signals are reported to SMU.EMM unit. An alarm for the upper and lower bound is supported in the SMU.EMM unit.

## Power Management System (PMS)

### EVRSTAT

#### EVR Status Register

(002C<sub>H</sub>)Reset Value: [Table 305](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	UVDD M	UVSB	UVPRE	OVDD M	OVSB	OVPRE	EVRCMOD			SDVK	SWDL VL	EVR33 SHHV	EVR33 SHLV	EVRCSS HHV	EVRCSS HLV
r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSTS WD	RST33	RSTC		0	EVR33 VOK	SYNCLK	UVSW D	UV33	UVC	OVSW D	OV33	EVR33	OVC	EVRC	
rh	rh	rh	r		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
EVRC	0	rh	<b>EVRC status</b> This bit is set if the internal EVRC regulator is currently active. EVRC is activated if HWCFG[2] pin level is latched high during start-up phase. 0 <sub>B</sub> EVRC is inactive. 1 <sub>B</sub> EVRC is active.
OVC	1	rh	<b>VDD Over-voltage event flag</b> This bit is set if VDD secondary voltage monitor recognizes a over-voltage event. An alarm is raised to the SMU and the status bit remains set until violation disappears. 0 <sub>B</sub> No Over-voltage condition or event active. 1 <sub>B</sub> VDD Over-voltage condition event indication as configured in EVROVMON / EVRMONCTRL register.
EVR33	2	rh	<b>EVR33 status</b> This bit is set if the internal EVR33 LDO regulator is active. EVR33 is activated if HWCFG[1] pin level is latched high during start-up phase. 0 <sub>B</sub> EVR33 is inactive. 1 <sub>B</sub> EVR33 is active.
OV33	3	rh	<b>VDDP3 Over-voltage event flag</b> This bit is set if VDDP3 secondary voltage monitor recognizes a over-voltage event. An alarm is raised to the SMU and the status bit remains set until violation disappears. 0 <sub>B</sub> No over-voltage condition or event active. 1 <sub>B</sub> VDDP3 Over-voltage event indication as configured in EVROVMON / EVRMONCTRL register.
OVSWD	4	rh	<b>VEXT Over-voltage event flag</b> This bit is set if VEXT secondary voltage monitor recognizes an over-voltage event. An alarm is raised to the SMU and the status bit remains set until violation disappears. 0 <sub>B</sub> No over-voltage condition or event active. 1 <sub>B</sub> VEXT Over-voltage event indication as configured in EVROVMON / EVRMONCTRL register.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>UVC</b>	5	rh	<p><b>VDD Under-voltage event flag</b></p> <p>This bit is set if VDD secondary voltage monitor recognizes a under-voltage event. An alarm is raised to the SMU and the status bit remains set until violation disappears.</p> <p>0<sub>B</sub> No under-voltage condition or event active. 1<sub>B</sub> VDD Under-voltage event indication as configured in EVRUVMON / EVRMONCTRL register.</p>
<b>UV33</b>	6	rh	<p><b>VDDP3 Under-voltage event flag</b></p> <p>This bit is set if VDDP3 secondary voltage monitor recognizes a under-voltage event. An alarm is raised to the SMU and the status bit remains set until violation disappears.</p> <p>0<sub>B</sub> No under-voltage condition or event active. 1<sub>B</sub> VDDP3 Under-voltage event indication as configured in EVRUVMON / EVRMONCTRL register.</p>
<b>UVSWD</b>	7	rh	<p><b>VEXT Under-voltage event flag</b></p> <p>This bit is set if VEXT secondary voltage monitor recognizes an under-voltage event. An alarm is raised to the SMU and the status bit remains set until violation disappears.</p> <p>0<sub>B</sub> No under-voltage condition or event active. 1<sub>B</sub> VEXT Under-voltage event indication as configured in EVRUVMON / EVRMONCTRL register.</p>
<b>SYNCLCK</b>	8	rh	<p><b>EVRC Synchronization Input Locked status(sd_sync_in_locked_o)</b></p> <p>This bitfield indicates the current synchronization status of EVRC SMPS regulator to external DCDCSYNCI input signal. When the EVRC switching frequency/ edge is locked to the synchronization input, the SYNCLCK bit is set to HIGH indicating the locked state. When the synchronization is lost owing to frequency deviations beyond MAXDEV or the feature is disabled via SYNCEN, the SYNCLCK bit is set to LOW.</p> <p>This EVRC Synchronization status is indicated in EVRSDSTAT0.SYNCLCK status bits.</p> <p>0<sub>B</sub> EVRC regulator runs on internal configured switching frequency and is not currently synchronized to external DCDCSYNCI input signal. 1<sub>B</sub> EVRC regulator switching frequency and VGATE output edge is currently synchronized to external DCDCSYNCI input signal.</p>
<b>EVR33VOK</b>	9	rh	<p><b>EVR33 Regulator Voltage OK status</b></p> <p>This bit is set after the soft ramp-up time of the EVR33 voltage OK ramp detector has elapsed and is not based on the measured VDDP3 voltage at the end of ramp-phase..</p> <p>0<sub>B</sub> EVR33 ramp-up time has not elapsed. 1<sub>B</sub> EVR33 ramp-up time has elapsed.</p>
<b>RSTC</b>	13	rh	<p><b>EVRC Reset Trigger</b></p> <p>0<sub>B</sub> No cold reset trigger signal is active after spike filter and core VDD voltage output is above the selected reset trim value. 1<sub>B</sub> A cold reset trigger signal is active after spike filter and core VDD voltage output is below the selected reset trim value.</p>

## Power Management System (PMS)

Field	Bits	Type	Description
<b>RST33</b>	14	rh	<b>EVR33 Reset Trigger</b> 0 <sub>B</sub> No cold reset trigger signal is active after spike filter and 3.3 V VDDP3 voltage output is above the selected reset trim value. 1 <sub>B</sub> A cold reset trigger signal is active after spike filter and 3.3 V VDDP3 voltage output is below the selected reset trim value.
<b>RSTSVD</b>	15	rh	<b>EVR SWD Reset Trigger</b> 0 <sub>B</sub> No cold reset trigger signal is active after spike filter and VEXT voltage input is above the selected reset trim value. 1 <sub>B</sub> A cold reset trigger signal is active after spike filter and VEXT voltage input is below the selected reset trim value.
<b>EVRCSHLV</b>	16	rh	<b>Short to ground</b> This bit is set if a short condition to ground has been detected. The measured EVRC output is below the operational supply range and the upper controller limits are reached. The feature is supported only during closed loop operation or EVRCMOD = 00b. 0 <sub>B</sub> No short to ground detected on VDD rail. 1 <sub>B</sub> Short to ground detected on VDD rail.
<b>EVRCSHHV</b>	17	rh	<b>Short to supply</b> This bit is set if a short condition to supply has been detected. The measured EVRC output exceeds the allowed supply range and the lower controller limits are reached. The feature is supported only during closed loop operation or EVRCMOD = 00b. 0 <sub>B</sub> No short to supply detected on VDD rail. 1 <sub>B</sub> Short to supply detected on VDD rail.
<b>EVR33SHLV</b>	18	rh	<b>Short to ground</b> This bit is set if a short condition to ground has been detected. The measured EVR33 output is below the operational supply range and the lower gate drive threshold voltage driving P ch. MOSFET is reached. 0 <sub>B</sub> No short to ground detected on VDDP3 rail. 1 <sub>B</sub> Short to ground detected on VDDP3 rail.
<b>EVR33SHHV</b>	19	rh	<b>Short to supply</b> This bit is set if a short condition to supply has been detected. The measured EVR33 output exceeds the allowed supply range and the upper gate drive threshold voltage driving P ch. MOSFET is reached. 0 <sub>B</sub> No short to supply detected on VDDP3 rail. 1 <sub>B</sub> Short to supply detected on VDDP3 rail.
<b>SWDLVL</b>	20	rh	<b>VEXT External Supply Level Status</b> This bit indicates that the VEXT voltage has dropped below ~4 V to indicate EVRC parameter switch to differentiate 5V or 3.3V external supply. A hysteresis of ~120 mV is implemented on this detector. 0 <sub>B</sub> VEXT external supply is above the threshold. 1 <sub>B</sub> VEXT external supply is below the threshold.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>SDVOK</b>	21	rh	<p><b>EVRC Regulator Voltage OK status</b></p> <p>This bit is set by the EVRC voltage OK detector to indicate that the new regulator output value has been reached. This bit is reset incase EVRTRIM, SDVOUTSEL or SDVOUTTRIM values are adapted to scale core voltage and is set when the new output setpoint is reached. This bit is also reset incase droop compensation is requested before a load jump event. A time out period of x us shall be waited when polling SDVOK bit.</p> <ul style="list-style-type: none"> <li><math>0_B</math> EVRC regulator setpoint voltage has not been reached.</li> <li><math>1_B</math> EVRC regulator setpoint voltage is reached and VDD voltage is ok.</li> </ul>
<b>EVRCMOD</b>	23:22	rh	<p><b>EVRC Mode</b></p> <p>This bit indicates the current operation mode of LC - PWM, LPM, STRT.</p> <ul style="list-style-type: none"> <li><math>00_B</math> SMPS Normal PWM Mode (PWM): The step-down converter is in normal operational closed loop state. Both Pch. MOSFET and Nch. MOSFET are being switched.</li> <li><math>01_B</math> SMPS Low Power Mode (LPM): The step-down converter is in low power state. Only Pch. MOSFET is being switched and Nch. MOSFET behaves like a diode.</li> <li><math>10_B</math> SMPS Start-up Mode (STRT): The step-down converter is in start-up phase. Only Pch. MOSFET is being switched and Nch. MOSFET behaves like a diode.</li> <li><math>11_B</math> EVRC is disabled.</li> </ul>
<b>OVPRE</b>	24	rh	<p><b>Pre Regulator VDDPD Over-voltage event flag</b></p> <p>This bit is set if VDDPD supply secondary voltage monitor recognizes an over-voltage event. An alarm is raised to the SMU.</p> <ul style="list-style-type: none"> <li><math>0_B</math> No over-voltage condition happened.</li> <li><math>1_B</math> VDDPD Over-voltage event indication as configured in EVROVMON2 register.</li> </ul>
<b>OVSB</b>	25	rh	<p><b>Standby Supply or VEVRSB Over-voltage event flag</b></p> <p>This bit is set if VEVRSB supply secondary voltage monitor recognizes an over-voltage event. An alarm is raised to the SMU.</p> <ul style="list-style-type: none"> <li><math>0_B</math> No over-voltage condition happened.</li> <li><math>1_B</math> VEVRSB Over-voltage event indication as configured in EVROVMON2 register.</li> </ul>
<b>OVDDM</b>	26	rh	<p><b>ADC VDDM Supply Over-voltage event flag</b></p> <p>This bit is set if VDDM ADC supply secondary voltage monitor recognizes an over-voltage event. An alarm is raised to the SMU.</p> <ul style="list-style-type: none"> <li><math>0_B</math> No over-voltage condition happened.</li> <li><math>1_B</math> VDDM Over-voltage event indication as configured in EVROVMON2 register.</li> </ul>
<b>UVPRE</b>	27	rh	<p><b>Pre Regulator VDDPD Under-voltage event flag</b></p> <p>This bit is set if VDDPD supply secondary voltage monitor recognizes an under-voltage event. An alarm is raised to the SMU.</p> <ul style="list-style-type: none"> <li><math>0_B</math> No under-voltage condition happened.</li> <li><math>1_B</math> VDDPD Under-voltage event indication as configured in EVRUVMON2 register.</li> </ul>

## Power Management System (PMS)

Field	Bits	Type	Description
UVSB	28	rh	<b>Standby Supply or VEVRSB Under-voltage event flag</b> This bit is set if VEVRSB supply secondary voltage monitor recognizes an under-voltage event. An alarm is raised to the SMU. 0 <sub>B</sub> No under-voltage condition happened. 1 <sub>B</sub> VEVRSB Under-voltage event indication as configured in EVRUVMON2 register.
UVDDM	29	rh	<b>ADC VDDM Supply Under-voltage event flag</b> This bit is set if VDDM ADC supply secondary voltage monitor recognizes an under-voltage event. An alarm is raised to the SMU. 0 <sub>B</sub> No under-voltage condition happened. 1 <sub>B</sub> VDDM Under-voltage event indication as configured in EVRUVMON2 register.
0	12:10, 31:30	r	<b>Reserved</b> Read as 0.

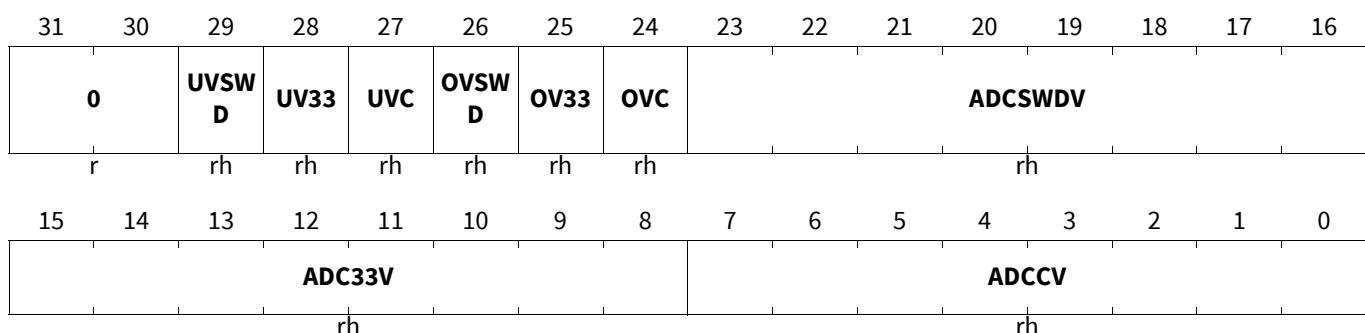
**Table 305 Reset Values of EVRSTAT**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

### EVR Primary ADC Status Register

#### EVRADCSTAT

**EVR Primary ADC Status Register (0034<sub>H</sub>) LVD Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
ADCCV	7:0	rh	<b>ADC VDD Core Voltage Conversion Result</b> This bit field contains the last filtered conversion result of the ADC measurement of the VDD / EVRC supply by the Primary Monitor. $VIN = [0.7125 + (ADCCV * LSB)] V$ LSB = 5 mV Eg. 1.25 V = 6C

## Power Management System (PMS)

Field	Bits	Type	Description
<b>ADC33V</b>	15:8	rh	<b>ADC VDDP3 Voltage Conversion Result</b> This bit field contains the last filtered conversion result of the ADC measurement of the VDDP3 / EVR33 supply by the Primary Monitor. $VIN = [0.9375 + (ADC33V * LSB)] V$ LSB = 15 mV Eg. 3.3 V = 9E
<b>ADCSWDV</b>	23:16	rh	<b>ADC VEXT Supply Conversion Result</b> This bit field contains the last filtered conversion result of the ADC measurement of the external VEXT (3.3V / 5V) supply by the Primary Monitor. $VIN = [1.050 + (ADCSWDV * LSB)] V$ LSB = 20 mV Eg. 5 V = C6
<b>OVC</b>	24	rh	<b>EVRC Regulator or VDD Over-voltage event flag</b> This bit is set if VDD primary voltage monitor recognizes a over-voltage event. An alarm is raised to the HSM and SMU. 0 <sub>B</sub> No over-voltage condition happened. 1 <sub>B</sub> VDD Over-voltage event indication as configured in HSMOVMON register.
<b>OV33</b>	25	rh	<b>EVR33 Regulator or VDDP3 Over-voltage event flag</b> This bit is set if VDDP3 primary voltage monitor recognizes a over-voltage event. An alarm is raised to the HSM and SMU. 0 <sub>B</sub> No over-voltage condition happened. 1 <sub>B</sub> VDDP3 Over-voltage event indication as configured in HSMOVMON register.
<b>OVSWD</b>	26	rh	<b>Supply Watchdog (SWD) or VEXT Over-voltage event flag</b> This bit is set if VEXT primary voltage monitor recognizes an over-voltage event. An alarm is raised to the HSM and SMU. 0 <sub>B</sub> No over-voltage condition happened. 1 <sub>B</sub> VEXT Over-voltage event indication as configured in HSMOVMON register.
<b>UVC</b>	27	rh	<b>EVRC Regulator or VDD Under-voltage event flag</b> This bit is set if VDD primary voltage monitor recognizes a under-voltage event. An alarm is raised to the HSM and SMU. 0 <sub>B</sub> No under-voltage condition happened. 1 <sub>B</sub> VDD Under-voltage event indication as configured in HSMUVMON register.
<b>UV33</b>	28	rh	<b>EVR33 Regulator or VDDP3 Under-voltage event flag</b> This bit is set if VDDP3 primary voltage monitor recognizes a under-voltage event. An alarm is raised to the HSM and SMU. 0 <sub>B</sub> No under-voltage condition happened. 1 <sub>B</sub> VDDP3 Under-voltage event indication as configured in HSMUVMON register.

## Power Management System (PMS)

Field	Bits	Type	Description
UVSWD	29	rh	<b>Supply Watchdog (SWD) or VEXT Under-voltage event flag</b> This bit is set if VEXT primary voltage monitor recognizes an under-voltage event. An alarm is raised to the HSM and SMU. 0 <sub>B</sub> No under-voltage condition happened. 1 <sub>B</sub> VEXT Under-voltage event indication as configured in HSMUVMON register.
0	31:30	r	<b>Reserved</b> Read as 0.

### EVR Reset Control Register

The **EVRRSTCON** register allows the activation/deactivation of the primary monitor under-voltage resets for the external supply and the generated EVR33 and EVRC voltages. The respective reset threshold trim values are also configured in this register

#### EVRRSTCON

##### EVR Reset Control Register

(003C<sub>H</sub>)

Reset Value: [Table 307](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SLCK	BPRST SWDO FF	RSTS WDOF F	BPRST 33OFF	RST33 OFF	BPRST COFF	RSTC OFF								
r	rw	w	rw	w	rw	w	rw								
RSTSVDTRIM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RST33TRIM								RSTCTRIM							
rw															

Field	Bits	Type	Description
RSTCTRIM	7:0	rw	<b>VDD Supply Reset Trim Value</b> This bit field selects the hard reset generation level of VDD supply rail. This bit field is trimmed by Firmware. $RSTCTRIM = [(VDDx - 712.5 \text{ mV}) / \text{LSB}]$ $VDDPRIUV = 712.5 \text{ mV} + \text{LSB} * RSTCTRIM$ (signed value) LSB = 5 mV
RST33TRIM	15:8	rw	<b>VDDP3 Supply Reset Trim Value</b> This bit field selects the hard reset generation level of VDDP3 supply rail. This bit field is trimmed by Firmware. $RST33TRIM = [(VDDx - 937.5 \text{ mV}) / \text{LSB}]$ $VDDP3PRIUV = 937.5 \text{ mV} + \text{LSB} * RST33TRIM + \text{LSB} * RST33PTRIM$ (signed value) LSB = 15 mV

## Power Management System (PMS)

Field	Bits	Type	Description
<b>RSTSWDTRIM</b>	23:16	rw	<p><b>VEXT Supply Reset Trim Value</b></p> <p>This bit field selects the hard reset generation level of the external VEXT supply rail. This bitfield is trimmed by Firmware.</p> $\text{RSTSWDTRIM} = [(VDDx - 1050 \text{ mV}) / \text{LSB}]$ $\text{VEXTPRIUV} = 1050 \text{ mV} + \text{LSB} * \text{RSTSWDTRIM}$ $\text{LSB} = 20 \text{ mV}$
<b>RSTCOFF</b>	24	rw	<p><b>VDD Reset Enable</b></p> <p>This bit can only be changed if bit BPRSTCOFF is set in parallel. RSTCOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application.</p> <p><math>0_B</math> A reset trigger signal is generated and forwarded to the SCU by primary monitor depending on the selected reset trim value.</p> <p><math>1_B</math> No reset trigger signal is generated and forwarded to the SCU by primary monitor depending on the selected reset trim value.</p>
<b>BPRSTCOFF</b>	25	w	<p><b>Bit Protection RSTCOFF</b></p> <p>Setting this bit enables that bit RSTCOFF can be changed in this write operation. This bit is read as zero.</p>
<b>RST33OFF</b>	26	rw	<p><b>VDDP3 Reset Enable</b></p> <p>This bit can only be changed if bit BPRST33OFF is set in parallel. The VDDP3 reset is disabled by application to support voltage drop up to nominal 3.0 V during cranking. RST33OFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application.</p> <p><math>0_B</math> A reset trigger signal is generated and forwarded to the SCU by primary monitor depending on the selected reset trim value.</p> <p><math>1_B</math> No reset trigger signal is generated and forwarded to the SCU by primary monitor depending on the selected reset trim value.</p>
<b>BPRST33OFF</b>	27	w	<p><b>Bit Protection RST33OFF</b></p> <p>Setting this bit enables that bit RST33OFF can be changed in this write operation. This bit read also as zero.</p>
<b>RSTSWDOFF</b>	28	rw	<p><b>VEXT Reset Enable</b></p> <p>This bit can only be changed if bit BPRSTSWDOFF is set in parallel. RSTSWDOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application.</p> <p><math>0_B</math> A reset trigger signal is generated and forwarded to the SCU by primary monitor depending on the selected reset trim value.</p> <p><math>1_B</math> No reset trigger signal is generated and forwarded to the SCU by primary monitor depending on the selected reset trim value.</p>
<b>BPRSTSWDOFF</b>	29	w	<p><b>Bit Protection RSTSWDOFF</b></p> <p>Setting this bit enables that bit RSTSWDOFF can be changed in this write operation.</p> <p>This bit is read as zero.</p>

## Power Management System (PMS)

Field	Bits	Type	Description
<b>SLCK</b>	30	rw	<p><b>HSM Security Lock</b></p> <p>If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011<sub>B</sub>). A set operation performed by any other master or software is ignored and the bit is kept as cleared.</p> <p>0<sub>B</sub> No lock active 1<sub>B</sub> Lock is active</p>
<b>0</b>	31	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 306 Access Mode Restrictions of EVRRSTCON sorted by descending priority**

Mode Name	Access Mode		Description
<b>SLCK = 0 and write 1 to BPRSTSWDOFF</b>	rw	RSTSWDOFF	
<b>SLCK = 0 and write 1 to BPRST33OFF</b>	rw	RST33OFF	
<b>SLCK = 0 and write 1 to BPRSTCOFF</b>	rw	RSTCOFF	
<b>SLCK = 0</b>	rw	RST33TRIM, RSTCTRIM, RSTSWDTRIM	
<b>SLCK = 0</b>	w	BPRST33OFF, BPRSTCOFF, BPRSTSWDOFF	
(default)	r	RST33OFF, RST33TRIM, RSTCOFF, RSTCTRIM, RSTSWDOFF, RSTSWDTRIM, SLCK	
	rX	BPRST33OFF, BPRSTCOFF, BPRSTSWDOFF	

**Table 307 Reset Values of EVRRSTCON**

Reset Type	Reset Value	Note
LVD Reset	0059 7F4A <sub>H</sub>	
Cold PORST	0059 7F4A <sub>H</sub>	
After SSW execution	005C 834B <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## Power Management System (PMS)

### EVR Reset Status Register

#### EVRRSTSTAT

#### EVR Reset Status Register

(0044<sub>H</sub>)

Reset Value: [Table 308](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			<b>RSTS WDOF F</b>	0	<b>RST33 OFF</b>	0	<b>RSTC OFF</b>								<b>RSTSVD</b>
r		rh	r	rh	r	rh	r								rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>RST33</b>								<b>RSTC</b>							
rh								rh							

Field	Bits	Type	Description
<b>RSTC</b>	7:0	rh	<b>VDD Supply Reset Value Status</b> This bit field indicates the actual cold PORST reset trim setpoint for core voltage supply rail used by the Primary monitors. The value is updated via <a href="#">EVRRSTCON.RSTCTRIM</a> register. RSTC = RSTCTRIM + RSTCPTRIM(signed value) RSTC range = 0 up to 255 $VDPRIUV = 712.5 \text{ mV} + \text{LSB} * \text{RSTC}$ LSB = 5 mV
<b>RST33</b>	15:8	rh	<b>VDDP3 Supply Reset Value Status</b> This bit field indicates the actual cold PORST reset trim setpoint for 3.3 V supply rail used by the Primary monitors. The value is updated via <a href="#">EVRRSTCON.RST33TRIM</a> register. RST33 = RST33TRIM + RST33PTRIM (signed value) RST33 range = 0 up to 255 $VDP3PRIUV = 937.5 \text{ mV} + \text{LSB} * \text{RST33}$ LSB = 15 mV
<b>RSTSVD</b>	23:16	rh	<b>VEXT Supply Reset Value Status</b> This bit field indicates the actual cold PORST reset trim setpoint for 5 V supply rail used by the Primary monitors. The value is updated via <a href="#">EVRRSTCON.RSTSVDTRIM</a> register. RSTSVD = RSTSVDTRIM+ RSTSVDPTRIM (signed value) RSTSVD range = 0 up to 255 $VEXTPRIUV = 1050 \text{ mV} + \text{LSB} * \text{RSTSVD}$ LSB = 20 mV
<b>RSTCOFF</b>	24	rh	<b>EVRC Reset Enable Status</b> The value is updated via <a href="#">EVRRSTCON.RSTCOFF</a> register bit. 0 <sub>B</sub> A cold PORST is triggered incase of VDD primary under-voltage event 1 <sub>B</sub> No cold PORST is generated incase of a primary under-voltage event.

## Power Management System (PMS)

Field	Bits	Type	Description
RST33OFF	26	rh	<b>EVR33 Reset Enable Status</b> The value is updated via <b>EVRRSTCON.RST33OFF</b> register bit. 0 <sub>B</sub> A cold PORST is triggered incase of VDDP3 primary under-voltage event 1 <sub>B</sub> No cold PORST is generated incase of a primary under-voltage event.
RSTSWD OFF	28	rh	<b>EVR SWD Reset Enable</b> The value is updated via <b>EVRRSTCON.RSTSWD OFF</b> register bit. 0 <sub>B</sub> A cold PORST is triggered incase of VEXT primary under-voltage event 1 <sub>B</sub> No cold PORST is generated incase of a primary under-voltage event.
0	25, 27, 31:29	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 308 Reset Values of EVRRSTSTAT**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

### EVR Trim Control Register

EVRTRIM and EVRRSTCON register may be used to generate voltage stress conditions to subject the modules to voltages beyond normal operating ranges.

#### EVRTRIM

##### EVR Trim Control Register

(004C<sub>H</sub>)

Reset Value: [Table 310](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	<b>SLCK</b>	<b>SDVOUTTRIM</b>					<b>0</b>	<b>EVR33VOUTTRIM</b>							
rh	rw						r								rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>SDVOUTSEL</b>								<b>EVR33VOUTSEL</b>							
rw								rw							

## Power Management System (PMS)

Field	Bits	Type	Description
<b>EVR33VOUTS EL</b>	7:0	rw	<p><b>EVR33 Regulator Output Voltage Target Value</b></p> <p>The VDDP3 output level of the EVR33 LDO regulator. The ramp-up completion to the new target value is indicated via EVRSTAT.EVR33VOK bit. The (EVR33VOUTSEL + EVR33VOUTTRIM) setpoint value shall be programmed between 0x24 and 0xDA for valid closed loop PID regulator function.</p> <p>3.3 V - 9E<sub>H</sub> - 158<sub>D</sub>  <math>EVR33VOUTSEL = [(VDDP3 - 937.5 \text{ mV}) / \text{LSB}]</math>  <math>VDDP3 = 937.5 \text{ mV} + \text{LSB} * EVR33VOUTSEL</math>  <math>\text{LSB} = 15 \text{ mV}</math></p>
<b>SDVOUTSEL</b>	15:8	rw	<p><b>EVRC Regulator Output Voltage Target Value</b></p> <p>The VDD output level of the Step down regulator.</p> <p>1.25 V - 6C - 108<sub>D</sub>  <math>SDVOUTSEL = [(VDD - 712.5 \text{ mV}) / \text{LSB}]</math>;  <math>VDD = 712.5 \text{ mV} + \text{LSB} * SDVOUTSEL</math>;  <math>\text{LSB} = 5 \text{ mV}</math>.</p> <p>This register bitfield requires a parameter update via EVRSDCTRL0.UP for transfer to EVRC SMPS shadow register. The reaching of the new target value is indicated via EVRSTAT.SDVOKE bit.</p>
<b>EVR33VOUTT RIM</b>	21:16	rw	<p><b>EVR33 Regulator Output Voltage Trim Value</b></p> <p>The 6 bit ADC BIST trimming value offset added to the EVR33 output level value installed by firmware from the flash.</p> <p>VDDP3 Setpoint = EVR33VOUTSEL + EVR33VOUTTRIM (signed value)  <math>EVR33VOUTTRIM \text{ RANGE} = -32 \text{ to } 31 \text{ LSB}</math>  <math>\text{LSB} = 15 \text{ mV}</math></p>
<b>SDVOUTTRIM</b>	29:24	rw	<p><b>EVRC Regulator Output Voltage Trim Value(vtrim_trim_i)</b></p> <p>The 6 bit ADC BIST trimming value offset added to the EVRC output level value installed by firmware from the flash. The reaching of the new setpoint is indicated via EVRSTAT.SDVOKE bits</p> <p>VDD Setpoint = SDVOUTSEL + SDVOUTTRIM (signed value)  <math>SDVOUTTRIM \text{ RANGE} = -32 \text{ to } 31 \text{ LSB}</math>  <math>\text{LSB} = 5 \text{ mV}</math></p> <p>This register bitfield requires a parameter update via EVRSDCTRL0.UP for transfer to SMPS shadow register.</p>
<b>SLCK</b>	30	rw	<p><b>HSM Security Lock</b></p> <p>If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011<sub>B</sub>). A set operation performed by any other master or software is ignored and the bit is kept as cleared.</p> <p>0<sub>B</sub> No lock active      1<sub>B</sub> Lock is active</p>

## **Power Management System (PMS)**

Field	Bits	Type	Description
LCK	31	rh	<p><b>Lock Status</b></p> <p>This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <p><math>0_B</math> The register is unlocked and can be updated  <math>1_B</math> The register is locked and cannot be updated</p>
0	23:22	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 309 Access Mode Restrictions of EVRTRIM sorted by descending priority**

<b>Mode Name</b>	<b>Access Mode</b>		<b>Description</b>
<b>SLCK</b> = 0 and <b>LCK</b> = 0	rw	EVR33VOUTSEL, EVR33VOUTTRIM, SDVOUTSEL, SDVOUTTRIM	
(default)	r	EVR33VOUTSEL, EVR33VOUTTRIM, SDVOUTSEL, SDVOUTTRIM, SLCK	

**Table 310** Reset Values of **EVRTRIM**

Reset Type	Reset Value	Note
After SSW execution	0000 6C9E <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## EVR Trim Status Register

EVRTRIMSTAT

## EVR Trim Status Register

(0050<sub>II</sub>)

### **Reset Value: Table 311**

Field	Bits	Type	Description
<b>EVR33VOUTS EL</b>	7:0	rh	<b>EVR33 Regulator Output Voltage Target Value</b> This bitfield indicates EVR33 output target value as configured in EVTRIM.EVR33VOUTSEL.
<b>SDVOUTSEL</b>	15:8	rh	<b>EVRC Regulator Output Voltage Target Value</b> This bit field indicates the EVRC output level of the Step down regulator as configured in EVTRIM.SDVOUTSEL. (vosel_target_o)

## Power Management System (PMS)

Field	Bits	Type	Description
<b>EVR33VOUTT RIM</b>	21:16	rh	<b>EVR33 Regulator Output Voltage Trim Value</b> This bit field indicates the 6 bit ADC BIST trimming value offset added to the EVR33 output level value installed by firmware from flash configuration sector if production trimming is required.
<b>SDVOUTTRIM</b>	29:24	rh	<b>EVRC Regulator Output Voltage Trim Value(vtrim_trim_o)</b> This bit field indicates the 5 bit ADC BIST trimming value offset added to the EVRC output level value installed by firmware from flash configuration sector as configured in EVTRIM.SDVOUTTRIM.
<b>0</b>	23:22, 31:30	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 311 Reset Values of EVRTRIMSTAT**

Reset Type	Reset Value	Note
LVD Reset	0000 6C9E <sub>H</sub>	
Cold PORST	0000 6C9E <sub>H</sub>	

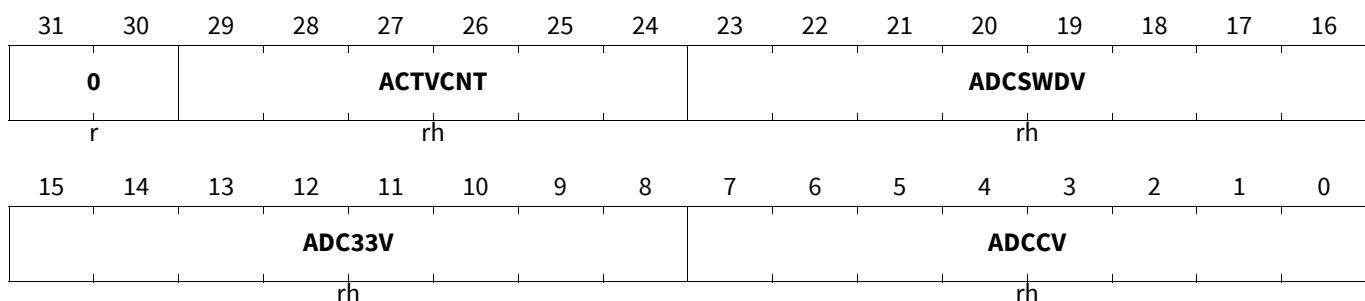
## EVR Secondary ADC Status Register 1

### EVRMONSTAT1

#### EVR Secondary ADC Status Register 1

(0060<sub>H</sub>)

Reset Value: [Table 312](#)



Field	Bits	Type	Description
<b>ADCCV</b>	7:0	rh	<b>VDD Supply Secondary ADC Conversion Result</b> This bit field contains the last conversion result of the ADC measurement of the VDD / EVRC supply by the Secondary Monitor. This bitfield is updated if secondary over- or under-voltage monitoring is activated via EVRMONCTRL.EVRCxxMOD. $VIN = [\text{LSB} * (\text{ADCx}-1)]$ ; Ideal LSB = 5.7692 mV Full Range : 1465 mV E.g. 1.25 V = DA

## Power Management System (PMS)

Field	Bits	Type	Description
<b>ADC33V</b>	15:8	rh	<b>VDDP3 Supply Secondary ADC Conversion Result</b> This bit field contains the last conversion result of the ADC measurement of the VDDP3 / EVR33 supply by the Secondary Monitor. This bitfield is updated if secondary over- or under-voltage monitoring is activated via EVRMONCTRL.EVR33xxMOD. VIN = [LSB * (ADCx-1)] ; Ideal LSB = 15.00 mV Full Range : 3810 mV E.g. 3.30 V = DD
<b>ADCSWDV</b>	23:16	rh	<b>VEXT Supply Secondary ADC Conversion Result</b> This bit field contains the last conversion result of the ADC measurement of the external VEXT (3.3V / 5V) supply by the Secondary Monitor. This bitfield is updated if secondary over- or under-voltage monitoring is activated via EVRMONCTRL.SWDxxMOD. VIN = [LSB * (ADCx-1)] ; LSB = 23.077 mV Full Range : 5861 mV E.g. 5.01 V = DA 3.3 V = 90
<b>ACTVCNT</b>	29:24	rh	<b>Secondary Monitor Activity Counter</b> This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations. The total number of conversions ConvTot = $\sum$ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
<b>0</b>	31:30	r	<b>Reserved</b> Read as 0.

**Table 312 Reset Values of EVRMONSTAT1**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

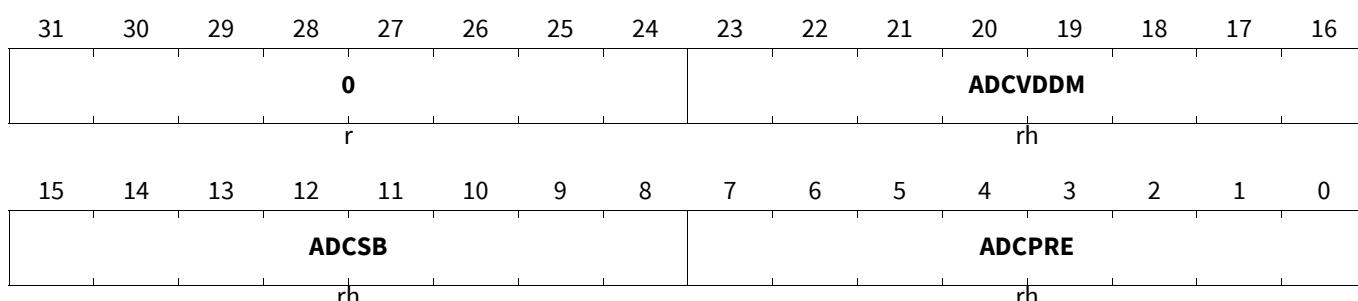
## EVR Secondary ADC Status Register 2

### EVRMONSTAT2

#### EVR Secondary ADC Status Register 2

(0064<sub>H</sub>)

Reset Value: [Table 313](#)



## Power Management System (PMS)

Field	Bits	Type	Description
<b>ADCPRE</b>	7:0	rh	<b>VDDPD Supply Secondary ADC Conversion Result</b> This bit field contains the last conversion result of the ADC measurement of the VDDPD supply by the Secondary Monitor. This bitfield is updated if secondary over- or under-voltage monitoring is activated via EVRMONCTRL.PRExxMOD. VIN = [LSB * (ADCx-1)] ; Ideal LSB = 5.7692 mV Full Range : 1465 mV E.g. 1.25 V = DA
<b>ADCSB</b>	15:8	rh	<b>VEVRSB Supply Secondary ADC Conversion Result</b> This bit field contains the last conversion result of the ADC measurement of the external VEVRSB (3.3V / 5V) standby supply by the Secondary Monitor. This bitfield is updated if secondary over- or under-voltage monitoring is activated via EVRMONCTRL.SBxxMOD. VIN = [LSB * (ADCx-1)] ; Ideal LSB = 23.077 mV Full Range : 5861 mV E.g. 5.01 V = DA 3.0 V = 90
<b>ADCVDDM</b>	23:16	rh	<b>VDDM Supply Secondary ADC Conversion Result</b> This bit field contains the last conversion result of the ADC measurement of the VDDM ADC supply by the Secondary Monitor. This bitfield is updated if secondary over- or under-voltage monitoring is activated via EVRMONCTRL.VDDMxxMOD. VIN = [LSB * (ADCx-1)] ; Ideal LSB = 23.077 mV Full Range : 5861 mV E.g. 5.01 V = DA <sub>D</sub> 3.0 V = 90 <sub>D</sub>
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

**Table 313 Reset Values of EVRMONSTAT2**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

### EVR Secondary Monitor Control Register

The default setting after reset is that over-voltage indication is notified via an SMU alarm when the over-voltage threshold is crossed in a lower to higher voltage transition. Overvoltage monitors use greater than equal compare if xOVMOD=01b or 11b and less than equal compare if xOVMOD=10b

The default setting after reset is that under-voltage indication is notified via an SMU alarm when the under-voltage threshold is crossed in a higher to lower voltage transition. Under voltage monitors use greater than equal compare if xUVMOD=01b and less than equal compare if xUVMOD=10b or 11b

It can be configured in EVRMONCTRL register to generate an interrupt when the over- under-voltage thresholds are crossed in either direction. This may be used to notify when the violation condition disappears with respect

## Power Management System (PMS)

to secondary voltage monitoring. Interrupt is generated on low to high transition of the EVRSTAT monitoring bits incase of xOVMOD=01b or 10b and interrupt is generated on any transition incase of xOVMOD=11b.

### EVRMONCTRL

#### EVR Secondary Monitor Control Register

(0068<sub>H</sub>)Reset Value: [Table 315](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SLCK		0					SBUVMOD	SWDUVMOD	SBOVMOD	SWDOVMOD				
r	rw		r					rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VDDMUVMOD	EVR33UVMOD	VDDMOVMOD	EVR33OVMOD	PREUVMOD	EVRCUVMOD	PREOVMOD	EVRCOVMOD								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
<b>EVRCOVMOD</b>	1:0	rw	<b>VDD Over-voltage monitoring mode</b> Incase both EVRCOVMOD = 00 <sub>B</sub> & EVRCUVMOD = 00 <sub>B</sub> , then ADC conversion for the respective supply rail does not take place. 00 <sub>B</sub> Over-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted. 01 <sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used. 10 <sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used. 11 <sub>B</sub> An over-voltage event is triggered when the threshold is crossed in either direction. Greater than or equal compare is used.
<b>PREOVMOD</b>	3:2	rw	<b>EVRPR or VDDPD Over-voltage monitoring mode</b> Incase both PREOVMOD = 00 <sub>B</sub> & PREUVMOD = 00 <sub>B</sub> , then ADC conversion for the respective supply rail does not take place. 00 <sub>B</sub> Over-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted. 01 <sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used. 10 <sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used. 11 <sub>B</sub> An over-voltage event is triggered when the threshold is crossed in either direction. Greater than or equal compare is used.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>EVRCUVMOD</b>	5:4	rw	<p><b>VDD Under-voltage monitoring mode</b></p> <p>Incase both EVRCOVMOD = 00<sub>B</sub> &amp; EVRCUVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Under-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in either direction. Less than or equal compare is used.</li> </ul>
<b>PREUVMOD</b>	7:6	rw	<p><b>EVRPR or VDDPD Under-voltage monitoring mode</b></p> <p>Incase both PREOVMOD = 00<sub>B</sub> &amp; PREUVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Under-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in either direction. Less than or equal compare is used.</li> </ul>
<b>EVR330VMOD</b>	9:8	rw	<p><b>VDDP3 Supply Over-voltage monitoring mode</b></p> <p>Incase both EVR330VMOD = 00<sub>B</sub> &amp; EVR33UVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Over-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in either direction. Greater than or equal compare is used.</li> </ul>

## Power Management System (PMS)

Field	Bits	Type	Description
<b>VDDMOVMOD</b>	11:10	rw	<p><b>VDDM ADC Supply Over-voltage monitoring mode</b>            Incase both VDDMOVMOD = 00<sub>B</sub> &amp; VDDMUVMOD = 00<sub>B</sub>, then ADC conversion for the VDDM supply rail continues to run as used for ADC function.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Over-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in either direction. Greater than or equal compare is used.</li> </ul>
<b>EVR33UVMOD</b>	13:12	rw	<p><b>VDDP3 Supply Under-voltage monitoring mode</b>            Incase both EVR33OVMOD = 00<sub>B</sub> &amp; EVR33UVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Under-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in either direction. Less than or equal compare is used.</li> </ul>
<b>VDDMUVMOD</b>	15:14	rw	<p><b>VDDM ADC Supply Under-voltage monitoring mode</b>            Incase both VDDMOVMOD = 00<sub>B</sub> &amp; VDDMUVMOD = 00<sub>B</sub>, then ADC conversion for the VDDM supply rail continues to run as used for ADC function.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Under-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in either direction. Less than or equal compare is used.</li> </ul>

## Power Management System (PMS)

Field	Bits	Type	Description
<b>SWDOVMOD</b>	17:16	rw	<p><b>VEXT Over-voltage monitoring mode</b></p> <p>Incase both SWDOVMOD = 00<sub>B</sub> &amp; SWDUMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Over-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in either direction. Greater than or equal compare is used.</li> </ul>
<b>SBOVMOD</b>	19:18	rw	<p><b>EVR Standby Supply or VEVRSB Over-voltage monitoring mode</b></p> <p>Incase both SBOVMOD = 00<sub>B</sub> &amp; SBUVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Over-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in either direction. Greater than or equal compare is used.</li> </ul>
<b>SWDUMOD</b>	21:20	rw	<p><b>VEXT Under-voltage monitoring mode</b></p> <p>Incase both SWDOVMOD = 00<sub>B</sub> &amp; SWDUMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Under-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in either direction. Less than or equal compare is used.</li> </ul>

## Power Management System (PMS)

Field	Bits	Type	Description
SBUVMOD	23:22	rw	<p><b>EVR Standby Supply or VEVRSB Under-voltage monitoring mode</b></p> <p>Incase both SBOVMOD = 00<sub>B</sub> &amp; SBUVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <p>00<sub>B</sub> Under-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</p> <p>01<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</p> <p>10<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</p> <p>11<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in either direction. Less than or equal compare is used.</p>
SLCK	30	rw	<p><b>HSM Security Lock</b></p> <p>If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011<sub>B</sub>). A set operation performed by any other master or software is ignored and the bit is kept as cleared.</p> <p>0<sub>B</sub> No lock active</p> <p>1<sub>B</sub> Lock is active</p>
0	29:24, 31	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 314 Access Mode Restrictions of EVRMONCTRL sorted by descending priority**

Mode Name	Access Mode		Description
SLCK = 0	rw	EVR33OVMOD, EVR33UVMOD, EVRCOVMOD, EVRCUVMOD, PREOVMOD, PREUVMOD, SBOVMOD, SBUVMOD, SWDOVMOD, SWDUVMOD, VDDMOVMOD, VDDMUVMOD	
(default)	r	EVR33OVMOD, EVR33UVMOD, EVRCOVMOD, EVRCUVMOD, PREOVMOD, PREUVMOD, SBOVMOD, SBUVMOD, SLCK, SWDOVMOD, SWDUVMOD, VDDMOVMOD, VDDMUVMOD	

**Table 315 Reset Values of EVRMONCTRL**

Reset Type	Reset Value	Note
LVD Reset	00A5 A5A5 <sub>H</sub>	

## Power Management System (PMS)

**Table 315 Reset Values of EVRMONCTRL (cont'd)**

Reset Type	Reset Value	Note
Cold PORST	00A5 A5A5 <sub>H</sub>	
After SSW execution	00A5 A5A5 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVR Secondary Monitor Filter Register

The assertion of alarm takes place when xFIL consecutive values are violating the threshold. Incase one of the values is not violating the threshold , the spike filter is reset. For renewed assertion of the alarm to take place, a repeated set of xFIL consecutive values violating the threshold are required.

#### EVRMONFILT

**EVR Secondary Monitor Filter Register (0070<sub>H</sub>)** Reset Value: [Table 317](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SLCK	CLRFI L			0				SBFIL			SWDFIL			
r	rw	rw			r				rw			rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		VDDMFIL			EVR33FIL			PREFIL			EVRCFIL				
		rw			rw			rw			rw				

Field	Bits	Type	Description
EVRCFIL	3:0	rw	<b>VDD Secondary ADC Supply Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
PREFIL	7:4	rw	<b>VDDPD Secondary ADC Supply Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
EVR33FIL	11:8	rw	<b>VDDP3 Secondary ADC Supply Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
VDDMFIL	15:12	rw	<b>VDDM Secondary ADC Supply Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>SWDFIL</b>	19:16	rw	<b>VEXT Secondary ADC Supply Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
<b>SBFIL</b>	23:20	rw	<b>VEVRSB Secondary ADC Supply Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
<b>CLRFIL</b>	29	rw	<b>Clear all Spike Filters</b> To avoid spurious alarms during change of configuration or start-up, CLRFIL shall be set followed by alarm reconfiguration followed by activation of filter logic by clearing CLRFIL register bit. $0_B$ No effect $1_B$ All spike filters configured in EVRMONFILT register are reset. The xFIL configuration value remains as configured and continue to be used for adc filtration.
<b>SLCK</b>	30	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master ( $TAG = 000011_B$ ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. $0_B$ No lock active $1_B$ Lock is active
<b>0</b>	28:24, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 316 Access Mode Restrictions of EVRMONFILT sorted by descending priority**

Mode Name	Access Mode		Description
<b>SLCK = 0</b>	rw	CLRFIL, EVR33FIL, EVRCFIL, PREFIL, SBFIL, SWDFIL, VDDMFIL	
(default)	r	CLRFIL, EVR33FIL, EVRCFIL, PREFIL, SBFIL, SLCK, SWDFIL, VDDMFIL	

**Table 317 Reset Values of EVRMONFILT**

Reset Type	Reset Value	Note
LVD Reset	0000 0300 <sub>H</sub>	

## Power Management System (PMS)

**Table 317 Reset Values of EVRMONFILT** (cont'd)

Reset Type	Reset Value	Note
Cold PORST	0000 0300 <sub>H</sub>	
After SSW execution	0001 0301 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### PMS Interrupt Enable Register

#### PMSIEN

#### PMS Interrupt Enable Register

(0074<sub>H</sub>)

Reset Value: [Table 318](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SCRW DT	SCREC C	SCRRS T	SCRIN T	PINBW KP	PINAW KP	ESR1 WKP	ESR0 WKP	WUTW KP	0	SWDL VL	SYNCL CK	SDVO K	EVRC MOD	
r	rw	rw	rw	rw	rw	rw	rw	rw	rw	r	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				UVSB	OVSBD	UVDDM	OVDDM	UVPRE	OVPRE	UVC	OVC	UV33	OV33	UVSWD	OVSWD
r				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
<b>OVSWD</b>	0	rw	<b>OVSWD Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>UVSWD</b>	1	rw	<b>UVSWD Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>OV33</b>	2	rw	<b>OV33 Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>UV33</b>	3	rw	<b>UV33 Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>OVC</b>	4	rw	<b>OVC Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>UVC</b>	5	rw	<b>UVC Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>OVPRE</b>	6	rw	<b>OVPRE Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>UVPRE</b>	7	rw	<b>UVPRE Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>OVDDM</b>	8	rw	<b>OVDDM Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>UVDDM</b>	9	rw	<b>UVDDM Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>OVSB</b>	10	rw	<b>OVSB Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>UVSB</b>	11	rw	<b>UVSB Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>EVRCMOD</b>	16	rw	<b>EVRCMOD Interrupt enable</b> Interrupt triggered on a state change of EVRSTAT.EVRCMOD[0] bitfield. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>SDVOK</b>	17	rw	<b>SDVOK Interrupt enable</b> Interrupt triggered on EVRSTAT.SDVOK rising edge event. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>SYNCLCK</b>	18	rw	<b>SD SYNCLCK Interrupt enable</b> Interrupt triggered on a state change of EVRSTAT.SYNCLCK bitfield. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>SWDLVL</b>	19	rw	<b>SWDLVL Interrupt enable</b> Interrupt triggered on a state change of EVRSTAT.SWDLVL bitfield. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>WUTWKP</b>	21	rw	<b>WUTWKP Interrupt enable</b> Interrupt triggered on a WUTCNT underflow event. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>ESR0WKP</b>	22	rw	<b>ESR0WKP Interrupt enable</b> Interrupt triggered on a ESR0WKP event. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>ESR1WKP</b>	23	rw	<b>ESR1WKP Interrupt enable</b> Interrupt triggered on a ESR1WKP event. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>PINAWKP</b>	24	rw	<b>PINAWKP Interrupt enable</b> Interrupt triggered on a PINAWKP event. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>PINBWKP</b>	25	rw	<b>PINBWKP Interrupt enable</b> Interrupt triggered on a PINBWKP event. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>SCRINT</b>	26	rw	<b>SCRINT Interrupt enable</b> Interrupt triggered on a SCRINT event triggered by SCR to PMS to decode information in PMSWCR2.SCRINT register. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>SCRRST</b>	27	rw	<b>SCRRST Interrupt enable</b> Interrupt triggered by SCR to PMS on an internal SCR software reset. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>SCRECC</b>	28	rw	<b>SCRECC Interrupt enable</b> Interrupt triggered by SCR to PMS on an internal RAM double bit ECC error. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>SCRWDT</b>	29	rw	<b>SCRWDT Interrupt enable</b> Interrupt triggered by SCR to PMS on an internal SCR watchdog timeout error. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>0</b>	15:12, 20, 31:30	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 318 Reset Values of PMSIEN**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

## Power Management System (PMS)

### EVR Secondary Under-voltage Monitor Register

A programmable threshold with upper and lower voltage bounds can be defined in EVROVMON and EVRUVMON registers for monitoring EVRC and EVR33 regulator outputs. Gain and Offset corrected thresholds can be evaluated from datasheet VxxMON parameters

#### EVRUVMON

##### EVR Secondary Under-voltage Monitor Register (0078<sub>H</sub>)

Reset Value: [Table 320](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	<b>SLCK</b>			0											<b>SWDUVVAL</b>
r	rw			r											rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															<b>EVRCUVVAL</b>

rw

rw

Field	Bits	Type	Description
<b>EVRCUVVAL</b>	7:0	rw	<b>VDD Supply Secondary Monitor Under-voltage threshold</b> This field defines the under-voltage monitoring threshold level of the EVRC regulator output or VDD supply. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 5.7692 mV
<b>EVR33UVVAL</b>	15:8	rw	<b>VDDP3 Supply Secondary Monitor Under-voltage threshold</b> This field defines the under-voltage monitoring threshold level of the EVR33 regulator output or VDDP3 supply. Ideal Threshold = [(VIN / LSB) + 1]. Ideal LSB = 15.00 mV
<b>SWDUVVAL</b>	23:16	rw	<b>VEXT Supply Secondary Monitor Under-voltage threshold</b> This field defines the under-voltage threshold level of the external VEXT supply monitor. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 23.077 mV.
<b>SLCK</b>	30	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011 <sub>B</sub> ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. 0 <sub>B</sub> No lock active 1 <sub>B</sub> Lock is active
<b>0</b>	29:24, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System (PMS)

**Table 319 Access Mode Restrictions of EVRUVMON sorted by descending priority**

Mode Name	Access Mode		Description
SLCK = 0	rw	EVR33UVVAL, EVRCUVVAL, SWDUVVAL	
(default)	r	EVR33UVVAL, EVRCUVVAL, SLCK, SWDUVVAL	

**Table 320 Reset Values of EVRUVMON**

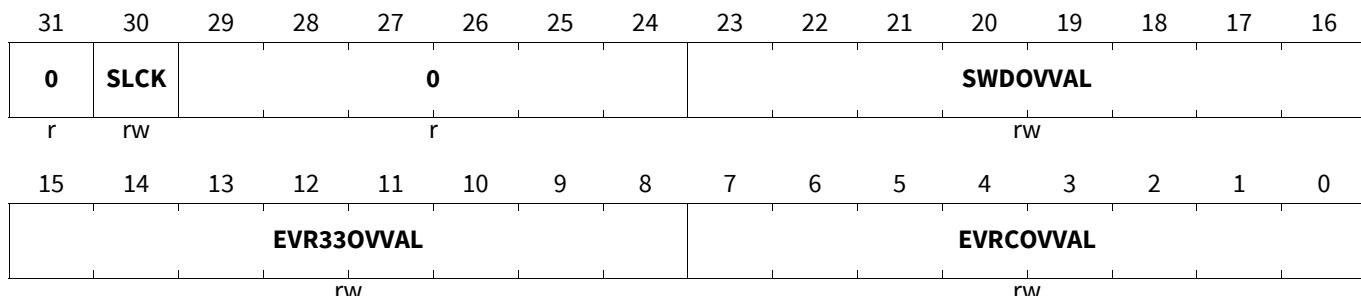
Reset Type	Reset Value	Note
LVD Reset	0075 A7B8 <sub>H</sub>	
Cold PORST	0075 A7B8 <sub>H</sub>	
After SSW execution	0075 A7B8 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVR Secondary Over-voltage Monitor Register

#### EVROVMON

#### EVR Secondary Over-voltage Monitor Register (007C<sub>H</sub>)

Reset Value: [Table 322](#)



Field	Bits	Type	Description
EVRCOVVAL	7:0	rw	<b>VDD Supply Secondary Monitor Over-voltage threshold</b> This field defines the over-voltage monitoring threshold level of the EVRC regulator output or VDD supply. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 5.7692 mV
EVR33OVVAL	15:8	rw	<b>VDDP3 Supply Secondary Monitor Over-voltage threshold</b> This field defines the over-voltage monitoring threshold level of the EVR33 regulator output or VDDP3 supply. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 15.00 mV
SWDOVVAL	23:16	rw	<b>VEXT Supply Secondary Monitor Over-voltage threshold</b> This field defines the over-voltage threshold level of the external VEXT supply monitor. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 23.077 mV.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>SLCK</b>	30	rw	<p><b>HSM Security Lock</b>            If this bit is set, all other bits in this register can no longer be written.            Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (<math>TAG = 000011_B</math>). A set operation performed by any other master or software is ignored and the bit is kept as cleared.</p> <p>0<sub>B</sub> No lock active            1<sub>B</sub> Lock is active</p>
<b>0</b>	29:24, 31	r	<p><b>Reserved</b>            Read as 0; should be written with 0.</p>

**Table 321 Access Mode Restrictions of EVROVMON sorted by descending priority**

Mode Name	Access Mode		Description
<b>SLCK = 0</b>	rw	EVR33OVVAL, EVRCOVVAL, SWDOVVAL	
(default)	r	EVR33OVVAL, EVRCOVVAL, SLCK, SWDOVVAL	

**Table 322 Reset Values of EVROVMON**

Reset Type	Reset Value	Note
LVD Reset	00FE FFFE <sub>H</sub>	
Cold PORST	00FE FFFE <sub>H</sub>	
After SSW execution	00FE FFFE <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## EVR Secondary Under-voltage Monitor Register 2

### EVRUVMON2

#### EVR Secondary Under-voltage Monitor Register 2(0080<sub>H</sub>)

Reset Value: [Table 324](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>0</b>	<b>SLCK</b>	<b>VDDMLVLSEL</b>					<b>SBUVVAL</b>								
r	rw			rw								rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>VDDMUVVAL</b>					<b>PREUVVAL</b>							rw			

## Power Management System (PMS)

Field	Bits	Type	Description
<b>PREUVVAL</b>	7:0	rw	<b>VDDPD Supply Secondary Monitor Under-voltage threshold</b> This field defines the under-voltage monitoring threshold level of the VDDPD supply or EVRPR output. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 5.7692 mV
<b>VDDMUWVAL</b>	15:8	rw	<b>VDDM Supply Secondary Monitor Under-voltage threshold</b> This field defines the under-voltage monitoring threshold level of the VDDM ADC supply. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 23.077 mV
<b>SBUVVAL</b>	23:16	rw	<b>VEVRSB Supply Secondary Monitor Under-voltage threshold</b> This field defines the under-voltage threshold level of the external VEVRSB (3.3V / 5V) standby supply monitor. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 23.077 mV.
<b>VDDMLVLSEL</b>	29:24	rw	<b>VDDM Level Select</b> This field defines the under-voltage monitoring threshold level required by EVADC / EDSADC modules to differentiate between 5 V or 3.3 V VDDM supply level to adjust analog behavior to the actual voltage level. The 6 MSB bits of the ADC result is compared against VDDMLVLSEL with 4 LSB hysteresis. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 92.308 mV
<b>SLCK</b>	30	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011 <sub>B</sub> ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. 0 <sub>B</sub> No lock active 1 <sub>B</sub> Lock is active
<b>0</b>	31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 323 Access Mode Restrictions of EVRUVMON2 sorted by descending priority**

Mode Name	Access Mode		Description
<b>SLCK = 0</b>	rw	PREUVVAL, SBUVVAL, VDDMLVLSEL, VDDMUWVAL	
(default)	r	PREUVVAL, SBUVVAL, SLCK, VDDMLVLSEL, VDDMUWVAL	

## Power Management System (PMS)

**Table 324 Reset Values of EVRUVMON2**

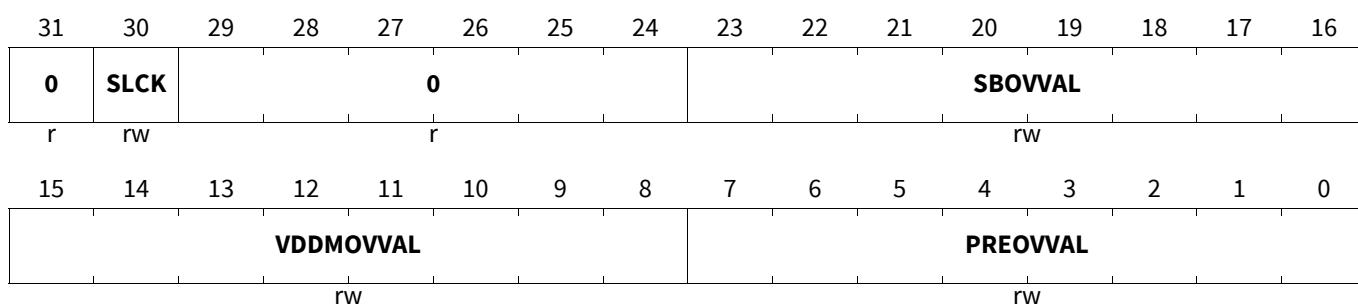
Reset Type	Reset Value	Note
LVD Reset	2A70 00BC <sub>H</sub>	
Cold PORST	2A70 00BC <sub>H</sub>	
After SSW execution	2A70 00BC <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVR Secondary Over-voltage Monitor Register 2

#### EVROVMON2

#### EVROVMON2 EVR Secondary Over-voltage Monitor Register 2 (0084<sub>H</sub>)

Reset Value: [Table 326](#)



Field	Bits	Type	Description
<b>PREOVAL</b>	7:0	rw	<b>VDDPD Supply Secondary Monitor Over-voltage threshold</b> This field defines the over-voltage monitoring threshold level of the VDDPD supply or EVRPR output. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 5.7692 mV
<b>VDDMOVVAL</b>	15:8	rw	<b>VDDM Supply Secondary Monitor Over-voltage threshold</b> This field defines the over-voltage monitoring threshold level of the VDDM ADC supply Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 23.077 mV
<b>SBOVAL</b>	23:16	rw	<b>VEVRSB Supply Secondary Monitor Over-voltage threshold</b> This field defines the over-voltage threshold level of the external VEVRSB (3.3V / 5V) standby supply monitor. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 23.077 mV
<b>SLCK</b>	30	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011 <sub>B</sub> ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. 0 <sub>B</sub> No lock active 1 <sub>B</sub> Lock is active

## Power Management System (PMS)

Field	Bits	Type	Description
0	29:24, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 325 Access Mode Restrictions of EVROVMON2 sorted by descending priority**

Mode Name	Access Mode	Description
SLCK = 0	rw	PREOVVAL, SBOVVAL, VDDMOVVAL
(default)	r	PREOVVAL, SBOVVAL, SLCK, VDDMOVVAL

**Table 326 Reset Values of EVROVMON2**

Reset Type	Reset Value	Note
LVD Reset	00FE FFFE <sub>H</sub>	
Cold PORST	00FE FFFE <sub>H</sub>	
After SSW execution	00FE FFFE <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## EVR Primary HSM Under-voltage Monitor Register

### HSMUVMON

#### EVR Primary HSM Under-voltage Monitor Register(0088<sub>H</sub>)

Reset Value: [Table 328](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>SLCK</b>	<b>HSMFIL</b>		<b>SWDO FF</b>	<b>EVR33 OFF</b>	<b>EVRC OFF</b>	<b>SWDUVVAL</b>									
rw		rw	rw	rw	rw										rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		<b>EVR33UVVAL</b>				<b>EVRCUVVAL</b>									rw

Field	Bits	Type	Description
<b>EVRCUVVAL</b>	7:0	rw	<b>VDD Supply Primary Monitor Alarm Under-voltage threshold</b> This field defines the under-voltage monitoring threshold level of the EVRC regulator output or VDD supply. EVRCUVVAL = [(VDDx - 712.5 mV) / LSB] LSB = 5 mV
<b>EVR33UVVAL</b>	15:8	rw	<b>VDDP3 Supply Primary Monitor Alarm Under-voltage threshold</b> This field defines the under-voltage monitoring threshold level of the EVR33 regulator output or VDDP3 supply. EVR33UVVAL = [(VDDx - 937.5 mV) / LSB] LSB = 15 mV

## Power Management System (PMS)

Field	Bits	Type	Description
<b>SWDUVVAL</b>	23:16	rw	<b>VEXT Supply Primary Monitor Alarm Under-voltage threshold</b> This field defines the under-voltage threshold level of the external VEXT supply monitor. $\text{SWDUVVAL} = [(VDDx - 1050 \text{ mV}) / \text{LSB}]$ LSB = 20 mV
<b>EVRCOFF</b>	24	rw	<b>VDD Primary Monitor UV Alarm Disable</b> $0_B$ A alarm trigger signal is generated and forwarded to the HSM by the EVRC block depending on the EVRCUVVAL configured value. $1_B$ No alarm trigger signal is generated and forwarded to the HSM by the EVRC block depending on the selected reset trim value.
<b>EVR33OFF</b>	25	rw	<b>VDDP3 Primary Monitor UV Alarm Disable</b> $0_B$ A alarm trigger signal is generated and forwarded to the HSM by the EVR33 block depending on the EVR33UVVAL configured value. $1_B$ No alarm trigger signal is generated and forwarded to the HSM by the EVR33 block depending on the selected reset trim value.
<b>SWDOFF</b>	26	rw	<b>VEXT Primary Monitor UV Alarm Disable</b> $0_B$ A alarm trigger signal is generated and forwarded to the HSM by the SWD block depending on the SWDUVVAL configured value. $1_B$ No alarm trigger signal is generated and forwarded to the HSM by the SWD block depending on the selected reset trim value.
<b>HSMFIL</b>	30:27	rw	<b>HSM Voltage Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to HSM.
<b>SLCK</b>	31	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011 <sub>B</sub> ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. $0_B$ No lock active $1_B$ Lock is active

Table 327 Access Mode Restrictions of **HSMUVMON** sorted by descending priority

Mode Name	Access Mode		Description
<b>SLCK = 0</b>	rw	EVR33OFF, EVR33UVVAL, EVRCOFF, EVRCUVVAL, HSMFIL, SWDOFF, SWDUVVAL	
(default)	r	EVR33OFF, EVR33UVVAL, EVRCOFF, EVRCUVVAL, HSMFIL, SLCK, SWDOFF, SWDUVVAL	

## Power Management System (PMS)

**Table 328 Reset Values of HSMUVMON**

Reset Type	Reset Value	Note
LVD Reset	005C 824D <sub>H</sub>	
Cold PORST	005C 824D <sub>H</sub>	
After SSW execution	005C 824D <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVR Primary HSM Over-voltage Monitor Register

#### HSMOVMON

#### EVR Primary HSM Over-voltage Monitor Register(008C<sub>H</sub>)

Reset Value: [Table 330](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>SLCK</b>			<b>0</b>		<b>SWDOFF</b>	<b>EVR33OFF</b>	<b>EVRCOFF</b>								<b>SWDOVVAL</b>
rw			r		rw	rw	rw								rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															<b>EVRCOVVAL</b>
															rw

Field	Bits	Type	Description
<b>EVRCOVVAL</b>	7:0	rw	<b>VDD Supply Primary Monitor Alarm Over-voltage threshold</b> This field defines the over-voltage monitoring threshold level of the EVRC regulator output or VDD supply. EVRCOVVAL = [(VDDx - 712.5 mV) / LSB] LSB = 5 mV
<b>EVR33OVVAL</b>	15:8	rw	<b>VDDP3 Supply Primary Monitor Alarm Over-voltage threshold</b> This field defines the over-voltage monitoring threshold level of the EVR33 regulator output or VDDP3 supply. EVR33OVVAL = [(VDDx - 937.5 mV) / LSB] LSB = 15 mV
<b>SWDOVVAL</b>	23:16	rw	<b>VEXT Supply Primary Monitor Alarm Over-voltage threshold</b> This field defines the over-voltage threshold level of the external VEXT supply monitor. SWDOVVAL = [(VDDx - 1050 mV) / LSB] LSB = 20 mV
<b>EVRCOFF</b>	24	rw	<b>VDD Primary Monitor OV Alarm Disable</b> 0 <sub>B</sub> A alarm trigger signal is generated and forwarded to the HSM by the EVRC block depending on the EVRCOVVAL configured value. 1 <sub>B</sub> No alarm trigger signal is generated and forwarded to the HSM by the EVRC block depending on the selected reset trim value.

## Power Management System (PMS)

Field	Bits	Type	Description
EVR33OFF	25	rw	<b>VDDP3 Primary Monitor OV Alarm Disable</b> 0 <sub>B</sub> A alarm trigger signal is generated and forwarded to the HSM by the EVR33 block depending on the EVR33OVVAL configured value. 1 <sub>B</sub> No alarm trigger signal is generated and forwarded to the HSM by the EVR33 block depending on the selected reset trim value.
SWDOFF	26	rw	<b>VEXT Primary Monitor OV Alarm Disable</b> 0 <sub>B</sub> A alarm trigger signal is generated and forwarded to the HSM by the SWD block depending on the SWDOVVAL configured value. 1 <sub>B</sub> No alarm trigger signal is generated and forwarded to the HSM by the SWD block depending on the selected reset trim value.
SLCK	31	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011 <sub>B</sub> ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. 0 <sub>B</sub> No lock active 1 <sub>B</sub> Lock is active
0	30:27	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 329 Access Mode Restrictions of HSMOVMON sorted by descending priority**

Mode Name	Access Mode		Description
SLCK = 0	rw	EVR33OFF, EVR33OVVAL, EVRCOFF, EVRCOVVAL, SWDOFF, SWDOVVAL	
(default)	r	EVR33OFF, EVR33OVVAL, EVRCOFF, EVRCOVVAL, SLCK, SWDOFF, SWDOVVAL	

**Table 330 Reset Values of HSMOVMON**

Reset Type	Reset Value	Note
LVD Reset	00E1 B586 <sub>H</sub>	
Cold PORST	00E1 B586 <sub>H</sub>	
After SSW execution	00E1 B586 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## Power Management System (PMS)

### EVR Oscillator Control Register

#### EVROSCCTRL

#### EVR Oscillator Control Register

(00A0<sub>H</sub>)

Reset Value: [Table 331](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>OSCT RIMEN</b>	0	<b>OSCTE MPOF FS</b>				0									<b>OSCFPTRIM</b>
rw	r	rw			r										rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				0											<b>OSCFTTRIM</b>
				r											rw

Field	Bits	Type	Description
<b>OSCFTTRIM</b>	5:0	rw	<b>Back-up Clock Fine Trim Value</b> This thermometer coded bit field contains information about the 100MHz OSC fine trimming. $f_{BACK\ ftrim} = [(OSCFTTRIM + (OSCFPTRIM)) * LSBFT] \text{ MHz}$ ; LSBFT = 110kHz Back-up Clock accuracy is documented in datasheet. It is recommended to wait 1 us after every fine trim step so that the clock source settles at the new frequency. fBACK ftrim value is saturated to range of 64. 00 <sub>H</sub> 0 MHz 1F <sub>H</sub> 3.65 MHz 3F <sub>H</sub> 7.3 MHz
<b>OSCFPTRIM</b>	21:16	rw	<b>OSC Fine Trim Signed Value</b> This bit field allows device individual trimming of the oscillator trim value during application. After updating the trim value, a waiting time of 1 us is required for the change to take effect.
<b>OSCTEMPOFF S</b>	29	rw	<b>Oscillator Temperature Offset Coefficient</b> This bitfield enables the centering function of the HPOSOC temperature coefficient to compensate for technology variations. 0 <sub>B</sub> Centering on. 1 <sub>B</sub> Centering off.
<b>OSCTRIMEN</b>	31	rw	<b>Dynamic Oscillator Trim Enable</b> Based on temperature, Oscillator can be trimmed. 0 <sub>B</sub> The Dynamic Oscillator Trim function is disabled/switched off. 1 <sub>B</sub> The Dynamic Oscillator Trim function is enabled.
0	15:6, 28:22, 30	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System (PMS)

**Table 331 Reset Values of EVROSCCTRL**

Reset Type	Reset Value	Note
LVD Reset	0000 001F <sub>H</sub>	
After SSW execution	2000 001F <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

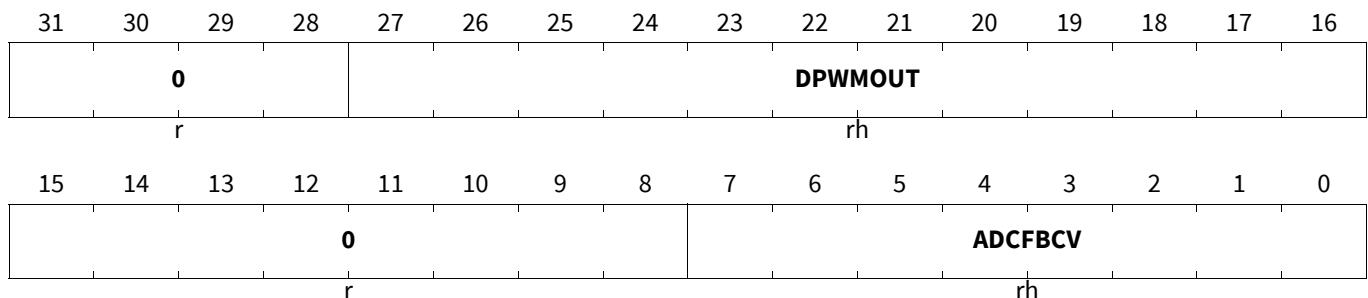
### EVR SD Status Register 0

**EVRSDSTAT0**

**EVR SD Status Register 0**

(00FC<sub>H</sub>)

**Reset Value: Table 332**



Field	Bits	Type	Description
<b>ADCFBCV</b>	7:0	rh	<b>Step Down Converter Core Voltage Feedback ADC Conversion Result</b> This bit field indicates the last ADC conversion result of the step down converter feedback ADC measuring VDD core voltage. $VIN = [LSB * (ADCFBCV - EVRTRIM.SDVOUTTRIM) + 0.7125] V$ ; LSB = 5 mV E.g. 1.20 V - 62 - 98
<b>DPWMOUT</b>	27:16	rh	<b>DPWM Control Output Status</b> This bit field reflects the actual PWM output of the controller provided to the external MOSFET switches.
<b>0</b>	15:8, 31:28	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 332 Reset Values of EVRSDSTAT0**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

## Power Management System (PMS)

### EVRC SD Control Register 0

#### EVRSDCTRL0

#### EVRC SD Control Register 0

(0108<sub>H</sub>)

Reset Value: [Table 334](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	UP	PGOFF	NGOF F												
rh	rwh	rw	rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Field	Bits	Type	Description
SDFREQSPRD	15:0	rw	<b>Frequency Spread Threshold(freqsp_coeff_i)</b> This bit field defines the additional frequency spread to the nominal EVRC regulator switching frequency during operation
SDFREQ	27:16	rw	<b>Regulator Switching Frequency or Over-sampling Factor(m0osfl_fact_i+m0osfh_fact_i)</b> This bit field configures the EVRC regulator switching frequency during closed loop operation. The switching frequency is equal to (100 MHz / (SDFREQ+1)) value. SDFREQ represents the corresponding over-sampling factor or clock cycles in a period. 037 <sub>H</sub> 1.82 MHz (100 MHz/(54+1)) SMPS switching frequency 07D <sub>H</sub> 0.8 MHz (100 MHz/(124+1)) SMPS switching frequency
NGOFF	28	rw	<b>NMOS level during OFF state(drvslo_ngoff_i)</b> This bit field configures the state of N ch. MOSFET driver during start-up and shut-down phases. 0 <sub>B</sub> TRISTATE 1 <sub>B</sub> LOW
PGOFF	29	rw	<b>PMOS level during OFF state(drvslo_pgoff_i)</b> This bitfield configures the state of Pch. MOSFET driver during start-up and shut-down phases. 0 <sub>B</sub> HIGH 1 <sub>B</sub> TRISTATE

## Power Management System (PMS)

Field	Bits	Type	Description
UP	30	rwh	<p><b>Update request for SMPS register values</b></p> <p>This bitfield triggers the update of the current register values from PMS-FPI EVRC registers to the local SMPS module registers.</p> <p>It shall be ensured that ALL EVRSDCTRLx and EVRSDCOEFFx registers have correct and coherent values across the various registers before the update request is issued. In case of singular register update, the other register values should match and be consistent. After a cold PORST, the UP bit is set as default reset value to ensure that the complete SMPS regulator parameter set is set back to its reset state. Consequently, the UP bit is reset and a read delivers 0. The parameter update via UP bit is not allowed in start-up and low power mode.</p> <p>0<sub>B</sub> No action is undertaken. 1<sub>B</sub> A new complete EVRC parameter set is transferred to the SMPS module. All EVRSDCTRLx and EVRSDCOEFFx register contents are transferred.</p>
LCK	31	rh	<p><b>Lock Status</b></p> <p>This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <p>0<sub>B</sub> The register is unlocked and can be updated 1<sub>B</sub> The register is locked and cannot be updated</p>

**Table 333 Access Mode Restrictions of EVRSDCTRL0 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	NGOFF, PGOFF, SDFREQ, SDFREQSPRD	
	rwh	UP	
(default)	r	NGOFF, PGOFF, SDFREQ, SDFREQSPRD	
	rh	UP	

**Table 334 Reset Values of EVRSDCTRL0**

Reset Type	Reset Value	Note
LVD Reset	F039 0001 <sub>H</sub>	
Cold PORST	F039 0001 <sub>H</sub>	
After SSW execution	F039 0001 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## Power Management System (PMS)

### EVRC SD Control Register 1

#### EVRSDCTRL1

#### EVRC SD Control Register 1

(010C<sub>H</sub>)

Reset Value: [Table 336](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	SYNCE N	0		MOSKIP		MOADCZB	MODEADBD		MOSOCOEFF						
rh	rw	r		rw		rw	rw		rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOTON								MOTOFF							
rw								rw							

Field	Bits	Type	Description
<b>MOTOFF</b>	7:0	rw	<b>Minimum Off Time(m0toff_mintof_i)</b> This bitfield configures the minimum off-time within one period in 100MHz clock cycle periods during closed loop operation.
<b>MOTON</b>	15:8	rw	<b>Minimum On Time(m0ton_minton_i)</b> This bitfield configures the minimum on-time within one period in 100MHz clock cycle periods during closed loop operation.
<b>MOSOCOEFF</b>	19:16	rw	<b>S0 coefficient(m0s0_coeff_i)</b> This bitfield indicates the S0 coefficient during closed loop operation.
<b>MODEADBD</b>	21:20	rw	<b>Dead Band(m0s0_deadbd_i)</b> This bitfield specifies the dead band to block the ADC ripple during closed loop operation.
<b>MOADCZB</b>	23:22	rw	<b>ADC Zero Bin(m0fcfg_adczb_i)</b> This bitfield specifies the zero error bin during closed loop operation. 00 <sub>B</sub> No compensation. 01 <sub>B</sub> 1/8 10 <sub>B</sub> 1/4 11 <sub>B</sub> 3/8
<b>MOSKIP</b>	27:24	rw	<b>Skip Pulse Threshold(m0skip_thres_i)</b> This bitfield specifies the threshold to detect a skip pulse condition during closed loop operation. (N-channel MOSFET).
<b>SYNCEN</b>	30	rw	<b>EVRC Synchronization input enable(synci0_en_i)</b> This bitfield enables the input synchronization logic of EVRC SMPS regulator. When set to 1, the DCDC will start to lock to the external synchronization input signal. This EVRC Synchronization status is indicated in EVRSTAT.SYNCLCK status bits. 0 <sub>B</sub> Synchronization of EVRC switching gate outputs to external input signal is disabled. 1 <sub>B</sub> Synchronization of EVRC switching gate outputs to external input signal is enabled.

## Power Management System (PMS)

Field	Bits	Type	Description
LCK	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
0	29:28	r	<b>Reserved</b> Read as 0; should be written with 0.

Table 335 Access Mode Restrictions of **EVRSCTRL1** sorted by descending priority

Mode Name	Access Mode		Description
LCK = 0	rw	M0ADCZB, M0DEADBD, M0S0COEFF, M0SKIP, M0TOFF, M0TON, SYNCEN	
(default)	r	M0ADCZB, M0DEADBD, M0S0COEFF, M0SKIP, M0TOFF, M0TON, SYNCEN	

Table 336 Reset Values of **EVRSCTRL1**

Reset Type	Reset Value	Note
LVD Reset	8669 0708 <sub>H</sub>	
Cold PORST	8669 0708 <sub>H</sub>	
After SSW execution	8669 0708 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## EVRC SD Coefficient Register 0

### EVRSDCOEFF0

#### EVRC SD Coefficient Register 0

(0148<sub>H</sub>)

Reset Value: [Table 338](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	MOSR MPCO EFFFR AC	M0S2V OSRC	M0S2V INSRC		M0S2COEFF		M0FGETCOEFF		M0SRMPCOEFF						
rh	rw	rw	rw		rw		rw		rw						rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								MOSKI PEN	MOSF RGET	MORA MPEN	MOS4E N	MOS3C LIP	MOS3E N	MOS2E N	MOSOE N
								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MOSOEN	0	rw	<b>S0 Enable(m0en_s0en_i)</b> This bitfield enables the fast-forward error term.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>M0S2EN</b>	1	rw	<b>S2 Enable(m0en_s2en_i)</b> This bitfield enables the digital reconstruction of the inductor current.
<b>M0S3EN</b>	2	rw	<b>S3 Enable(m0en_s3en_i)</b> This bitfield enables the integrator.
<b>M0S3CLIP</b>	3	rw	<b>S3 Clip(m0en_s3clip_i)</b> This bitfield specifies the clipping of the integrator state to negative values.
<b>M0S4EN</b>	4	rw	<b>S4 Enable(m0en_s4en_i)</b> This bitfield enables the double integrator branch.
<b>M0RAMPEN</b>	5	rw	<b>Ramp Enable(m0en_rampen_i)</b> This bitfield enables the artificial ramp in order to avoid instabilities at high duty cycles.
<b>M0SFRGET</b>	6	rw	<b>SFRGET(m0en_sfrget_i)</b> This bitfield enables the compensation of parasitic effects in the inductor current reconstruction.
<b>M0SKIPEN</b>	7	rw	<b>Skip Enable(m0en_skipen_i)</b> This bitfield enables the skip pulse logic.
<b>M0S3COEFF</b>	11:8	rw	<b>S3 Coefficient(m0s3_coeff_i)</b> Configuration register of S3 - integrator coefficient.
<b>M0S4COEFF</b>	15:12	rw	<b>S4 Coefficient(m0s4_coeff_i)</b> Configuration register of S4 - double integrator coefficient.
<b>M0SRMPCOEF F</b>	19:16	rw	<b>S Ramp Coefficient(m0srmp_coeff_i)</b> Configuration register of S Ramp - artificial ramp coefficient.
<b>M0FGETCOEF F</b>	23:20	rw	<b>S2 Forgetting Factor(m0fget_coeff_i)</b> This bitfield specifies the forgetting factor for compensation of parasitic effects.
<b>M0S2COEFF</b>	27:24	rw	<b>S2 Coefficient(m0s2_coeff_i)</b> Inductor current reconstruction coefficient.
<b>M0S2VINSRC</b>	28	rw	<b>S2 Vin Source(m0s2_vinsrc_i)</b> This bitfield specifies the source of the input voltage used for the inductor current reconstruction. 0 <sub>B</sub> The register value M0VIN is used. 1 <sub>B</sub> The FF-ADC counter value is used
<b>M0S2VOSRC</b>	29	rw	<b>S2 Vout Source(m0s2_vosrc_i)</b> This bitfield specifies the source of the output voltage used for the inductor current reconstruction. 0 <sub>B</sub> The register value M0VO is used. 1 <sub>B</sub> The FB-ADC counter value is used
<b>M0SRMPCOEF FFRAC</b>	30	rw	<b>S Ramp Fractional Coefficient</b> This bitfield specifies the S Ramp fractional coefficient. 0 <sub>B</sub> no fractional coefficient used. 1 <sub>B</sub> fractional coefficient 1/2 used (SRMP + 0.5).

## Power Management System (PMS)

Field	Bits	Type	Description
LCK	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated

**Table 337 Access Mode Restrictions of EVRSDCOEFF0 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	M0FGETCOEFF, M0RAMPEN, M0S0EN, M0S2COEFF, M0S2EN, M0S2VINSRC, M0S2VOSRC, M0S3CLIP, M0S3COEFF, M0S3EN, M0S4COEFF, M0S4EN, M0SFRGET, M0SKIPEN, M0SRMPCOEFF, M0SRMPCOEFFFRAC	
(default)	r	M0FGETCOEFF, M0RAMPEN, M0S0EN, M0S2COEFF, M0S2EN, M0S2VINSRC, M0S2VOSRC, M0S3CLIP, M0S3COEFF, M0S3EN, M0S4COEFF, M0S4EN, M0SFRGET, M0SKIPEN, M0SRMPCOEFF, M0SRMPCOEFFFRAC	

**Table 338 Reset Values of EVRSDCOEFF0**

Reset Type	Reset Value	Note
LVD Reset	B508 73B6 <sub>H</sub>	
Cold PORST	B508 73B6 <sub>H</sub>	
After SSW execution	B508 73B6 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVRC SD Coefficient Register 1

#### EVRSDCOEFF1

#### EVRC SD Coefficient Register 1

(014C<sub>H</sub>)

Reset Value: [Table 340](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	<b>M0S2COEFFFF RAC</b>	<b>M0S3COEFFFF RAC</b>	<b>MOVIN</b>												
rh	rw	rw	rw												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>MOVOUT</b>								<b>MOVOCFINC</b>				<b>MOVOCFLPF</b>			
rw								rw				rw			

## Power Management System (PMS)

Field	Bits	Type	Description
<b>M0VOCFLPF</b>	3:0	rw	<b>LPF Coefficient(m0vocf_lpf_i)</b> This bitfield reflects LPF coefficient used in the LPF applied to the FB-ADC counter value or the programmed register value. $y [k] = \{ y [k-1] * (1-a) \} + \{ x [k] * a \}$ ; $y [k]$ is filter output; $x [k]$ is ADC output $a = \{1 / (2 ^ LPF)\}$ . If LPF = 0, the filter output is the same as ADC output.
<b>M0VOCFINC</b>	7:4	rw	<b>Output Voltage Ramp Coefficient(m0vocf_inc_i)</b> This bitfield reflects increment for the output voltage ramp used in the inductor current reconstruction. Step applied to ramp = $2 ^ M0VOCFINC$ .
<b>M0VOUT</b>	15:8	rw	<b>Digital representation of the target voltage(m0vo_lb_i)</b> This bitfield can be used for the inductor current reconstruction instead of the FBADC value.
<b>MOVIN</b>	26:16	rw	<b>Digital representation of the input voltage(m0vinh_vin_i+m0vinl_vin_i))</b> This bitfield is used for the inductor current reconstruction instead of the FFADC value. Absolute value including ADC offset.
<b>M0S3COEFFFRAC</b>	28:27	rw	<b>S3 Fractional Coefficient</b> This bitfield specifies the S3 fractional integrator coefficient. 00 - no fractional coefficient used 01 ... fractional coefficient 1/4 used ( $S3 + 0.25$ ) 10 ... fractional coefficient 1/2 used ( $S3 + 0.5$ ) 11 ... fractional coefficient 3/4 used ( $S3 + 0.75$ )
<b>M0S2COEFFFRAC</b>	30:29	rw	<b>S2 Fractional Coefficient</b> This bitfield specifies the S2 fractional coefficient of the inductor current reconstruction coefficient. 00 - no fractional coefficient used 01 ... fractional coefficient 1/4 used ( $S2 + 0.25$ ) 10 ... fractional coefficient 1/2 used ( $S2 + 0.5$ ) 11 ... fractional coefficient 3/4 used ( $S2 + 0.75$ )
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated

## Power Management System (PMS)

**Table 339 Access Mode Restrictions of [EVRSDCOEFF1](#) sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	M0S2COEFFFRAC, M0S3COEFFFRAC, M0VIN, M0VOCFINC, M0VOCFLPF, M0VOUT	
(default)	r	M0S2COEFFFRAC, M0S3COEFFFRAC, M0VIN, M0VOCFINC, M0VOCFLPF, M0VOUT	

**Table 340 Reset Values of [EVRSDCOEFF1](#)**

Reset Type	Reset Value	Note
LVD Reset	A294 6C46 <sub>H</sub>	
Cold PORST	A294 6C46 <sub>H</sub>	
After SSW execution	A294 6C46 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## EVRC SD Control Register 2

### EVRSDCTRL2

**EVRC SD Control Register 2** (0110<sub>H</sub>) Reset Value: [Table 342](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	<b>EVRC MOD</b>	<b>0</b>													
rh	rw	r													rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		<b>0</b>			<b>LPLPFCOEFF</b>				<b>LPBNDWIDTH</b>			<b>LPBNDOFFSET</b>			rw
		r			rw				rw						

Field	Bits	Type	Description
<b>LPBNDOFFSET</b>	3:0	rw	<b>Low Power Mode Hysteresis OFFSET(lpnd_offset_i)</b> This bitfield defines the turn-on threshold in LP mode
<b>LPBNDWIDTH</b>	7:4	rw	<b>Low Power Mode Hysteresis Band Width(lpnd_width_i)</b> This bitfield defines the turn-on threshold in LP mode.
<b>LPLPFCOEFF</b>	11:8	rw	<b>Low Pass Filter Coefficient(lplpf_coeff_i)</b> This bit field configures the low pass filter coefficient for the setting of the turn-on threshold of the Sliding function. 0 <sub>H</sub> Fast Filter F <sub>H</sub> Slow Filter

## Power Management System (PMS)

Field	Bits	Type	Description
SDFREQLP	27:16	rw	<b>Regulator Over-sampling Factor(m1osfl_fact_i+m1osfh_fact_i)</b> This bitfield configures the EVRC regulator FB ADC sampling period during low power mode. The switching frequency is not constant. 037 <sub>H</sub> 1.82 MHz (100 MHz/55) 07D <sub>H</sub> 0.8 MHz (100 MHz/125) 0C8 <sub>H</sub> 0.5 MHz (100 MHz/200)
EVRCMOD	30	rw	<b>LPM or PWM EVRC Mode Activation</b> This bit switches operation mode between PWM and LPM mode. 0 <sub>B</sub> The step-down converter is in normal operational closed loop state (PWM). Both Pch. MOSFET and Nch. MOSFET are being switched. 1 <sub>B</sub> The step-down converter is in low power mode (LPM). Only Pch. MOSFET is being switched and Nch. MOSFET behaves like a diode.
LCK	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
0	15:12, 29:28	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 341 Access Mode Restrictions of EVRSDCTRL2 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	EVRCMOD, LPBNDOFFSET, LPBNDWIDTH, LPLPFCOEFF, SDFREQLP	
(default)	r	EVRCMOD, LPBNDOFFSET, LPBNDWIDTH, LPLPFCOEFF, SDFREQLP	

**Table 342 Reset Values of EVRSDCTRL2**

Reset Type	Reset Value	Note
LVD Reset	0036 033B <sub>H</sub>	
Cold PORST	0036 033B <sub>H</sub>	
After SSW execution	0036 033B <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## **Power Management System (PMS)**

## **EVRC SD Control Register 3**

EVRSDCTRL3

## **EVRC SD Control Register 3**

(0114<sub>H</sub>)

## **Reset Value: Table 343**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	<b>0</b>				<b>M1SKIP</b>			<b>M1ADCZB</b>	<b>M1DEADBD</b>			<b>M1S0COEFF</b>			
	r				rw			rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					<b>M1TON</b>					<b>M1TOFF</b>					
					rw					rw					

Field	Bits	Type	Description
<b>M1TOFF</b>	7:0	rw	<b>Minimum Off Time(m1toff_mintof_i)</b> This bitfield configures the minimum off-time within one period in 100MHz clock cycle periods during LP mode.
<b>M1TON</b>	15:8	rw	<b>Minimum On Time(m1ton_minton_i)</b> This bitfield configures the minimum on-time within one period in 100MHz clock cycle periods during LP mode.
<b>M1S0COEFF</b>	19:16	rw	<b>S0 coefficient(m1s0_coeff_i)</b> This bitfield indicates the S0 coefficient during LP mode.
<b>M1DEADBD</b>	21:20	rw	<b>Dead Band(m1s0_deadbd_i)</b> This bitfield specifies the dead band to block the ADC ripple during LP mode.
<b>M1ADCZB</b>	23:22	rw	<b>ADC Zero Bin(m1fcfg_adczb_i)</b> This bitfield specifies the zero error bin during LP mode. 00 <sub>B</sub> No compensation. 01 <sub>B</sub> 1/8 10 <sub>B</sub> 1/4 11 <sub>B</sub> 3/8
<b>M1SKIP</b>	27:24	rw	<b>Skip Pulse Threshold(m1skip_thres_i)</b> This bitfield is disabled in LPM mode as PFM applied by control itself.
<b>0</b>	31:28	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 343** Reset Values of **EVRSDCTRL3**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
LVD Reset	0B69 0810 <sub>H</sub>	
Cold PORST	0B69 0810 <sub>H</sub>	
After SSW execution	0B69 0810 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## Power Management System (PMS)

### EVRC SD Coefficient Register 2

#### EVRSDCOEFF2

#### EVRC SD Coefficient Register 2

(0150<sub>H</sub>)

Reset Value: [Table 344](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	<b>M1S2V OSRC</b>	<b>M1S2V INSRC</b>	<b>M1S2COEFF</b>				<b>M1FGETCOEFF</b>				<b>M1SRMPCOEFF</b>				
r	rw	rw	rw				rw				rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>M1S4COEFF</b>				<b>M1S3COEFF</b>				<b>M1SKI PEN</b>	<b>M1SF RGET</b>	<b>M1RA MPEN</b>	<b>M1S4E N</b>	<b>M1S3C LIP</b>	<b>M1S3E N</b>	<b>M1S2E N</b>	<b>M1SOE N</b>
rw				rw				rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
<b>M1S0EN</b>	0	rw	<b>S0 Enable(m1en_s0en_i)</b> This bitfield enables the fast-forward error term.
<b>M1S2EN</b>	1	rw	<b>S2 Enable(m1en_s2en_i)</b> This bitfield enables the digital reconstruction of the inductor current.
<b>M1S3EN</b>	2	rw	<b>S3 Enable(m1en_s3en_i)</b> This bitfield enables the integrator.
<b>M1S3CLIP</b>	3	rw	<b>S3 Clip(m1en_s3clip_i)</b> This bitfield specifies the clipping of the integrator state to negative values.
<b>M1S4EN</b>	4	rw	<b>S4 Enable(m1en_s4en_i)</b> This bitfield enables the double integrator branch.
<b>M1RAMPEN</b>	5	rw	<b>Ramp Enable(m1en_rampen_i)</b> This bitfield enables the artificial ramp in order to avoid instabilities at high duty cycles.
<b>M1SFRGET</b>	6	rw	<b>SFRGET(m1en_sfrget_i)</b> This bitfield enables the compensation of parasitic effects in the inductor current reconstruction.
<b>M1SKIPEN</b>	7	rw	<b>Skip Enable(m1en_skipen_i)</b> This bitfield enables the skip pulse logic.
<b>M1S3COEFF</b>	11:8	rw	<b>S3 Coefficient(m1s3_coeff_i)</b> Configuration register of S3 - integrator coefficient.
<b>M1S4COEFF</b>	15:12	rw	<b>S4 Coefficient(m1s4_coeff_i)</b> Configuration register of S4 - double integrator coefficient.
<b>M1SRMPCOEF F</b>	19:16	rw	<b>S Ramp Coefficient(m1srmp_coeff_i)</b> Configuration register of S Ramp - artificial ramp coefficient.
<b>M1FGETCOEF F</b>	23:20	rw	<b>S2 Forgetting Factor(m1fget_coeff_i)</b> This bitfield specifies the forgetting factor for compensation of parasitic effects.
<b>M1S2COEFF</b>	27:24	rw	<b>S2 Coefficient(m1s2_coeff_i)</b> Inductor current reconstruction coefficient.

## **Power Management System (PMS)**

Field	Bits	Type	Description
<b>M1S2VINSRC</b>	28	rw	<p><b>S2 Vin Source(m1s2_vinsrc_i)</b></p> <p>This bitfield specifies the source of the input voltage used for the inductor current reconstruction.</p> <p>0<sub>B</sub> The register value M1VIN is used. 1<sub>B</sub> The FF-ADC counter value is used</p>
<b>M1S2VOSRC</b>	29	rw	<p><b>S2 Vout Source(m1s2_vosrc_i)</b></p> <p>This bitfield specifies the source of the output voltage used for the inductor current reconstruction.</p> <p>0<sub>B</sub> The register value M1VO is used. 1<sub>B</sub> The FB-ADC counter value is used</p>
<b>0</b>	31:30	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 344 Reset Values of EVRSDCOEFF2**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
LVD Reset	3408 710E <sub>H</sub>	
Cold PORST	3408 710E <sub>H</sub>	
After SSW execution	3408 710E <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

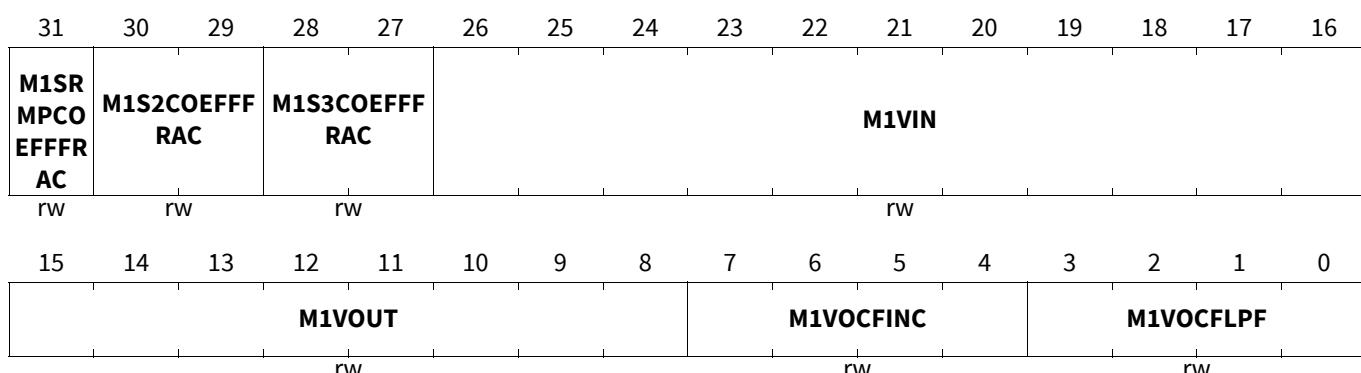
## **EVRC SD Coefficient Register 3**

EVRSRDCOEFF3

## **EVRC SD Coefficient Register 3**

(0154<sub>H</sub>)

## Reset Value: Table 345



<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>M1VOCFLPF</b>	3:0	rw	<p><b>LPF Coefficient(m1vocf_lpf_i)</b></p> <p>This bitfield reflects LPF coefficient used in the LPF applied to the FB-ADC counter value or the programmed register value.</p> $y [k] = \{ y [k-1] * (1-a) \} + \{ x [k] * a \}; y [k] \text{ is filter output; } x [k] \text{ is ADC output}$ $a = \{1 / (2 ^ \text{LPF})\}. \text{If LPF = 0, the filter output is the same as ADC output.}$

## Power Management System (PMS)

Field	Bits	Type	Description
<b>M1VOCFINC</b>	7:4	rw	<b>Output Voltage Ramp Coefficient(m1vocf_inc_i)</b> This bitfield reflects increment for the output voltage ramp used in the inductor current reconstruction. Step applied to ramp = $2^{\wedge} M1VOCFINC$ .
<b>M1VOUT</b>	15:8	rw	<b>Digital representation of the target voltage(m1vo_lb_i)</b> This bitfield can be used for the inductor current reconstruction instead of the FBADC value.
<b>M1VIN</b>	26:16	rw	<b>Digital representation of the input voltage(m1vinh_vin_i+m1vinl_vin_i)</b> This bitfield can be used for the inductor current reconstruction instead of the FFADC value. Absolute value including ADC offset.
<b>M1S3COEFFF RAC</b>	28:27	rw	<b>S3 Fractional Coefficient</b> This bitfield specifies the S3 fractional integrator coefficient. 00 - no fractional coefficient used 01 ... fractional coefficient 1/4 used ( $S3 + 0.25$ ) 10 ... fractional coefficient 1/2 used ( $S3 + 0.5$ ) 11 ... fractional coefficient 3/4 used ( $S3 + 0.75$ )
<b>M1S2COEFFF RAC</b>	30:29	rw	<b>S2 Fractional Coefficient</b> This bitfield specifies the S2 fractional coefficient of the inductor current reconstruction coefficient. 00 - no fractional coefficient used 01 ... fractional coefficient 1/4 used ( $S2 + 0.25$ ) 10 ... fractional coefficient 1/2 used ( $S2 + 0.5$ ) 11 ... fractional coefficient 3/4 used ( $S2 + 0.75$ )
<b>M1SRMPCOEF FFRAC</b>	31	rw	<b>S Ramp Fractional Coefficient</b> This bitfield specifies the S Ramp fractional coefficient. $0_B$ no fractional coefficient used $1_B$ fractional coefficient 1/2 used ( $SRMP + 0.5$ ).

**Table 345 Reset Values of EVRSDCOEFF3**

Reset Type	Reset Value	Note
LVD Reset	0294 6C44 <sub>H</sub>	
Cold PORST	0294 6C44 <sub>H</sub>	
After SSW execution	0294 6C44 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## Power Management System (PMS)

### EVRC SD Control Register 4

#### EVRSDCTRL4

#### EVRC SD Control Register 4

(0118<sub>H</sub>)

Reset Value: [Table 346](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16				
0				SDFREQST															
r								rw											
0								VOKCFG											
r								rw											

Field	Bits	Type	Description
VOKCFG	5:0	rw	<b>Voltage OK Circuit Configuration(vokcfg_config_i)</b> t.b.d.
SDFREQST	27:16	rw	<b>Regulator Switching Frequency or Over-sampling Factor(m2osfl_fact_i+m2osfh_fact_i)</b> This bit field configures the EVRC regulator switching frequency during closed loop start-up. The switching frequency is equal to (100 MHz / SDFREQ) value. SDFREQ represents the corresponding over-sampling factor. 037 <sub>H</sub> 1.82 MHz (100 MHz/55) SMPS switching frequency 07D <sub>H</sub> 0.8 MHz (100 MHz/125) SMPS switching frequency 0C8 <sub>H</sub> 0.5 MHz (100 MHz/200) SMPS switching frequency
0	15:6, 31:28	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 346 Reset Values of EVRSDCTRL4**

Reset Type	Reset Value	Note
LVD Reset	0036 0009 <sub>H</sub>	
Cold PORST	0036 0009 <sub>H</sub>	
After SSW execution	0036 0009 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## Power Management System (PMS)

### EVRC SD Control Register 5

#### EVRSDCTRL5

#### EVRC SD Control Register 5

(011C<sub>H</sub>)

Reset Value: [Table 347](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			0				M2SKIP			M2ADCZB	M2DEADBD			M2S0COEFF	
r				rw				rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					M2TON					M2TOFF					
					rw					rw					

Field	Bits	Type	Description
<b>M2TOFF</b>	7:0	rw	<b>Minimum Off Time(m2toff_mintof_i)</b> This bitfield configures the minimum off-time within one period in 100MHz clock cycle periods during closed loop operation.
<b>M2TON</b>	15:8	rw	<b>Minimum On Time(m2ton_minton_i)</b> This bitfield configures the minimum on-time within one period in 100MHz clock cycle periods during closed loop operation.
<b>M2S0COEFF</b>	19:16	rw	<b>S0 coefficient(m2s0_coeff_i)</b> This bitfield indicates the S0 coefficient during closed loop operation.
<b>M2DEADBD</b>	21:20	rw	<b>Dead Band(m2s0_deadbd_i)</b> This bitfield specifies the dead band to block the ADC ripple during closed loop operation.
<b>M2ADCZB</b>	23:22	rw	<b>ADC Zero Bin(m2fcfg_adczb_i)</b> This bitfield specifies the zero error bin during closed loop operation. 00 <sub>B</sub> No compensation. 01 <sub>B</sub> 1/8 10 <sub>B</sub> 1/4 11 <sub>B</sub> 3/8
<b>M2SKIP</b>	27:24	rw	<b>Skip Pulse Threshold(m2skip_thres_i)</b> This bitfield specifies the threshold to detect a skip pulse condition during closed loop operation. (N-channel MOSFET).
<b>0</b>	31:28	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 347 Reset Values of EVRSDCTRL5**

Reset Type	Reset Value	Note
LVD Reset	0B69 0808 <sub>H</sub>	
Cold PORST	0B69 0808 <sub>H</sub>	
After SSW execution	0B69 0808 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## Power Management System (PMS)

### EVRC SD Control Register 6

#### EVRSDCTRL6

#### EVRC SD Control Register 6

(0120<sub>H</sub>)

Reset Value: [Table 349](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>					<b>0</b>				<b>SINCHI</b>		<b>0</b>		<b>SINCLO</b>		
rh				r					rw		r		rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>SVOTH</b>								<b>SVINTH</b>							
rw								rw							

Field	Bits	Type	Description
<b>SVINTH</b>	7:0	rw	<b>Vin threshold to switch between SINCLO or SINCHI.(svinth_thres_i)</b> This bit field specifies the threshold to decide on the ramp-up increment during startup. If Vin is below the threshold, SINCLO is taken as ramp-up increment, else if Vin is equal or above the threshold, SINCHI is taken as ramp-up increment. The threshold is compared to the FF-ADC counter value, without offset.
<b>SVOTH</b>	15:8	rw	<b>Vout threshold to switch from open loop start-up to closed loop mode.(svoth_thres_i)</b> This bit field specifies the threshold to decide when to switch from open-loop mode to closed-loop mode during startup. If Vout is below the threshold, open-loop ramp up is executed. if Vout is equal or above the threshold, closed-loop PWM in start-up configuration is executed. The threshold is compared to the low pass filtered FB-ADC counter value, without offset. The switch happens only in one direction during startup and the system does not switch back into start-up mode even if threshold is crossed in other direction.
<b>SINCLO</b>	18:16	rw	<b>Increment for low input voltage.(sinc_sinclo_i)</b> This bitfield specifies the increment of the on-time during open-loop ramp-up during startup. If Vin is below the threshold (SVINTH), SINCLO is taken as ramp-up increment. if Vin is equal or above the threshold (SVINTH), SINCHI is taken as ramp-up increment
<b>SINCHI</b>	22:20	rw	<b>Increment for high input voltage.(sinc_sinchi_i)</b> This bitfield specifies the increment of the on-time during open-loop ramp-up during startup. If Vin is below the threshold (SVINTH), SINCLO is taken as ramp-up increment. if Vin is equal or above the threshold (SVINTH), SINCHI is taken as ramp-up increment
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
<b>0</b>	19, 30:23	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System (PMS)

**Table 348 Access Mode Restrictions of EVRSDCTRL6 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	SINCHI, SINCLO, SVINTH, SVOTH	
(default)	r	SINCHI, SINCLO, SVINTH, SVOTH	

**Table 349 Reset Values of EVRSDCTRL6**

Reset Type	Reset Value	Note
LVD Reset	8023 1C94 <sub>H</sub>	
Cold PORST	8023 1C94 <sub>H</sub>	
After SSW execution	8023 1C94 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVRC SD Coefficient Register 4

**EVRSDCOEFF4**

**EVRC SD Coefficient Register 4**

**(0158<sub>H</sub>)**

**Reset Value: Table 350**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	M2S2V OSRC	M2S2V INSRC		M2S2COEFF			M2FGETCOEFF			M2SRMPCOEFF					
r	rw	rw		rw			rw			rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	M2S4COEFF			M2S3COEFF			M2SKI PEN	M2SF RGET	M2RA MPEN	M2S4E N	M2S3C LIP	M2S3E N	M2S2E N	M2SOE N	
	rw			rw			rw	rw	rw	rw	rw	rw	rw	rw	

Field	Bits	Type	Description
<b>M2S0EN</b>	0	rw	<b>S0 Enable(m2en_s0en_i)</b> This bitfield enables the fast-forward error term.
<b>M2S2EN</b>	1	rw	<b>S2 Enable(m2en_s2en_i)</b> This bitfield enables the digital reconstruction of the inductor current.
<b>M2S3EN</b>	2	rw	<b>S3 Enable(m2en_s3en_i)</b> This bitfield enables the integrator.
<b>M2S3CLIP</b>	3	rw	<b>S3 Clip(m2en_s3clip_i)</b> This bitfield specifies the clipping of the integrator state to negative values.
<b>M2S4EN</b>	4	rw	<b>S4 Enable(m2en_s4en_i)</b> This bitfield enables the double integrator branch.
<b>M2RAMPEN</b>	5	rw	<b>Ramp Enable(m2en_rampen_i)</b> This bitfield enables the artificial ramp in order to avoid instabilities at high duty cycles.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>M2SFRGET</b>	6	rw	<b>SFRGET(m2en_sfrget_i)</b> This bitfield enables the compensation of parasitic effects in the inductor current reconstruction.
<b>M2SKIPEN</b>	7	rw	<b>Skip Enable(m2en_skipen_i)</b> This bitfield enables the skip pulse logic.
<b>M2S3COEFF</b>	11:8	rw	<b>S3 Coefficient(m2s3_coeff_i)</b> Configuration register of S3 - integrator coefficient.
<b>M2S4COEFF</b>	15:12	rw	<b>S4 Coefficient(m2s4_coeff_i)</b> Configuration register of S4 - double integrator coefficient.
<b>M2SRMPCOEF F</b>	19:16	rw	<b>S Ramp Coefficient(m2srmp_coeff_i)</b> Configuration register of S Ramp - artificial ramp coefficient.
<b>M2FGETCOEF F</b>	23:20	rw	<b>S2 Forgetting Factor(m2fget_coeff_i)</b> This bitfield specifies the forgetting factor for compensation of parasitic effects.
<b>M2S2COEFF</b>	27:24	rw	<b>S2 Coefficient(m2s2_coeff_i)</b> Inductor current reconstruction coefficient.
<b>M2S2VINSRC</b>	28	rw	<b>S2 Vin Source(m2s2_vinsrc_i)</b> This bitfield specifies the source of the input voltage used for the inductor current reconstruction. $0_B$ The register value M2VIN is used. $1_B$ The FF-ADC counter value is used
<b>M2S2VOSRC</b>	29	rw	<b>S2 Vout Source(m2s2_vosrc_i)</b> This bitfield specifies the source of the output voltage used for the inductor current reconstruction. $0_B$ The register value M2VO is used. $1_B$ The FB-ADC counter value is used
<b>0</b>	31:30	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 350 Reset Values of EVRSDCOEFF4**

Reset Type	Reset Value	Note
LVD Reset	1B08 22B6 <sub>H</sub>	
Cold PORST	1B08 22B6 <sub>H</sub>	
After SSW execution	1B08 22B6 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## **Power Management System (PMS)**

## **EVRC SD Coefficient Register 5**

EVRSDCOEFF5

## **EVRC SD Coefficient Register 5**

(015C<sub>H</sub>)

### **Reset Value: Table 351**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>M2SR</b> <b>MPCO</b> <b>EFFFR</b> <b>AC</b>	<b>M2S2COEFFF</b> RAC	<b>M2S3COEFFF</b> RAC								<b>M2VIN</b>					
rw	rw	rw								rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>M2VOUT</b>						<b>M2VOCFINC</b>				<b>M2VOCFLPF</b>					
rw						rw				rw				rw	

Field	Bits	Type	Description
M2VOCFLPF	3:0	rw	<p><b>LPF Coefficient(m2vocf_lpf_i)</b></p> <p>This bitfield reflects LPF coefficient used in the LPF applied to the FB-ADC counter value or the programmed register value.</p> $y [k] = \{ y [k-1] * (1-a) \} + \{ x [k] * a \}; y [k] \text{ is filter output; } x [k] \text{ is ADC output}$ <p><math>a = \{1 / (2 ^ \text{LPF})\}</math>. If LPF = 0, the filter output is the same as ADC output.</p>
M2VOCFINC	7:4	rw	<p><b>Output Voltage Ramp Coefficient(m2vocf_inc_i)</b></p> <p>This bitfield reflects the increment for the output voltage ramp used in the inductor current reconstruction.</p> <p>Step applied to ramp = <math>2 ^ \text{M2VOCFINC}</math>.</p>
M2VOUT	15:8	rw	<p><b>Digital representation of the target voltage(m2vo_lb_i)</b></p> <p>This bitfield can be used for the inductor current reconstruction instead of the FBADC value.</p>
M2VIN	26:16	rw	<p><b>Digital representation of the input voltage(m2vinh_vin_i+m2vinl_vin_i)</b></p> <p>This bitfield can be used for the inductor current reconstruction instead of the FFADC value. Absolute value including ADC offset.</p>
M2S3COEFFF RAC	28:27	rw	<p><b>S3 Fractional Coefficient</b></p> <p>This bitfield specifies the S3 fractional integrator coefficient.</p> <p>00 - no fractional coefficient used      01 ... fractional coefficient 1/4 used (<math>S3 + 0.25</math>)      10 ... fractional coefficient 1/2 used (<math>S3 + 0.5</math>)      11 ... fractional coefficient 3/4 used (<math>S3 + 0.75</math>)</p>
M2S2COEFFF RAC	30:29	rw	<p><b>S2 Fractional Coefficient</b></p> <p>This bitfield specifies the S2 fractional coefficient of the inductor current reconstruction coefficient.</p> <p>00 - no fractional coefficient used      01 ... fractional coefficient 1/4 used (<math>S2 + 0.25</math>)      10 ... fractional coefficient 1/2 used (<math>S2 + 0.5</math>)      11 ... fractional coefficient 3/4 used (<math>S2 + 0.75</math>)</p>

## Power Management System (PMS)

Field	Bits	Type	Description
<b>M2SRMPCOEFF FFRAC</b>	31	rw	<b>S Ramp Fractional Coefficient</b> This bitfield specifies the S Ramp fractional coefficient. $0_B$ no fractional coefficient used $1_B$ fractional coefficient 1/2 used (SRMP + 0.5).

**Table 351 Reset Values of EVRSDCOEFF5**

Reset Type	Reset Value	Note
LVD Reset	$0294\ 6C46_H$	
Cold PORST	$0294\ 6C46_H$	
After SSW execution	$0294\ 6C46_H$	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## EVRC SD Control Register 7

### EVRSDCTRL7

#### EVRC SD Control Register 7

( $0124_H$ )

Reset Value: [Table 353](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>			<b>0</b>			<b>SYNCDIVFAC</b>					<b>DRVSPR</b>				
rh			r			rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						<b>DRVSLOMOD</b>	<b>E</b>			<b>DRVPCBF</b>		<b>DRVNI</b>			
						rw				rw		rw			

Field	Bits	Type	Description
<b>DRVNI</b>	1:0	rw	<b>Selection of N-driver current</b> Adjustable driver strength of the N driver current $00_B$ 1/4 $01_B$ 1/2 $10_B$ 3/4 $11_B$ 1
<b>DRVPCBF</b>	3:2	rw	<b>P-Driver Current Boost Factor(drvp_strgth_i)</b> Adjustable boost factor for the P driver current $00_B$ 9 / 7 $01_B$ 9 / 5 $10_B$ 9 / 4 $11_B$ 9 / 3

## Power Management System (PMS)

Field	Bits	Type	Description																																
<b>DRV_P</b>	7:4	rw	<p><b>P-Driver Current(drvp_strgth_i)</b></p> <p>Base drive current of the P-channel MOSFET when driven with 3.3V / 5V.</p> <table> <tr><td>0<sub>H</sub></td><td>5,3 mA / 7,8 mA</td></tr> <tr><td>1<sub>H</sub></td><td>6,3 mA / 9,4 mA</td></tr> <tr><td>2<sub>H</sub></td><td>7,4 mA / 11 mA</td></tr> <tr><td>3<sub>H</sub></td><td>8,4 mA / 12,5 mA</td></tr> <tr><td>4<sub>H</sub></td><td>10,5 mA / 15,6 mA</td></tr> <tr><td>5<sub>H</sub></td><td>12,6 mA / 18,7 mA</td></tr> <tr><td>6<sub>H</sub></td><td>14,7 mA / 21,8 mA</td></tr> <tr><td>7<sub>H</sub></td><td>17,8 mA / 26,4 mA</td></tr> <tr><td>8<sub>H</sub></td><td>20,9 mA / 31 mA</td></tr> <tr><td>9<sub>H</sub></td><td>25 mA / 37,1 mA</td></tr> <tr><td>A<sub>H</sub></td><td>29,1 mA / 43,2 mA</td></tr> <tr><td>B<sub>H</sub></td><td>35,3 mA / 52,3 mA</td></tr> <tr><td>C<sub>H</sub></td><td>41,4 mA / 61,4 mA</td></tr> <tr><td>D<sub>H</sub></td><td>49,6 mA / 73,4 mA</td></tr> <tr><td>E<sub>H</sub></td><td>58,8 mA / 87 mA</td></tr> <tr><td>F<sub>H</sub></td><td>69,9 mA / 103,5 mA</td></tr> </table>	0 <sub>H</sub>	5,3 mA / 7,8 mA	1 <sub>H</sub>	6,3 mA / 9,4 mA	2 <sub>H</sub>	7,4 mA / 11 mA	3 <sub>H</sub>	8,4 mA / 12,5 mA	4 <sub>H</sub>	10,5 mA / 15,6 mA	5 <sub>H</sub>	12,6 mA / 18,7 mA	6 <sub>H</sub>	14,7 mA / 21,8 mA	7 <sub>H</sub>	17,8 mA / 26,4 mA	8 <sub>H</sub>	20,9 mA / 31 mA	9 <sub>H</sub>	25 mA / 37,1 mA	A <sub>H</sub>	29,1 mA / 43,2 mA	B <sub>H</sub>	35,3 mA / 52,3 mA	C <sub>H</sub>	41,4 mA / 61,4 mA	D <sub>H</sub>	49,6 mA / 73,4 mA	E <sub>H</sub>	58,8 mA / 87 mA	F <sub>H</sub>	69,9 mA / 103,5 mA
0 <sub>H</sub>	5,3 mA / 7,8 mA																																		
1 <sub>H</sub>	6,3 mA / 9,4 mA																																		
2 <sub>H</sub>	7,4 mA / 11 mA																																		
3 <sub>H</sub>	8,4 mA / 12,5 mA																																		
4 <sub>H</sub>	10,5 mA / 15,6 mA																																		
5 <sub>H</sub>	12,6 mA / 18,7 mA																																		
6 <sub>H</sub>	14,7 mA / 21,8 mA																																		
7 <sub>H</sub>	17,8 mA / 26,4 mA																																		
8 <sub>H</sub>	20,9 mA / 31 mA																																		
9 <sub>H</sub>	25 mA / 37,1 mA																																		
A <sub>H</sub>	29,1 mA / 43,2 mA																																		
B <sub>H</sub>	35,3 mA / 52,3 mA																																		
C <sub>H</sub>	41,4 mA / 61,4 mA																																		
D <sub>H</sub>	49,6 mA / 73,4 mA																																		
E <sub>H</sub>	58,8 mA / 87 mA																																		
F <sub>H</sub>	69,9 mA / 103,5 mA																																		
<b>DRVSLOMODE</b>	9:8	rw	<p><b>Switching Configuration(drvslo_mode_i)</b></p> <p>This bitfield configures the type of switching.</p> <table> <tr><td>00<sub>B</sub></td><td>Nominal mode</td></tr> <tr><td>01<sub>B</sub></td><td>B=C mode</td></tr> <tr><td>10<sub>B</sub></td><td>Hard Switching mode</td></tr> <tr><td>11<sub>B</sub></td><td>Reserved</td></tr> </table>	00 <sub>B</sub>	Nominal mode	01 <sub>B</sub>	B=C mode	10 <sub>B</sub>	Hard Switching mode	11 <sub>B</sub>	Reserved																								
00 <sub>B</sub>	Nominal mode																																		
01 <sub>B</sub>	B=C mode																																		
10 <sub>B</sub>	Hard Switching mode																																		
11 <sub>B</sub>	Reserved																																		
<b>DRVSPR</b>	23:16	rw	<b>Spare bits(drvspr_x_i)</b>																																
<b>SYNDIVFAC</b>	26:24	rw	<p><b>Switching frequency division factor for external synchronisation(synco_divfac_i)</b></p> <p>This bit field defines the divider factor for the SMPS switching output to generate DCDCSYNCO output to synchronize external regulator to the internal EVRC regulator. The signal is routed to pin if enabled via PMSWCR5.DCDCSYNCO bit.</p> <p>All other combinations are reserved.</p> <p>000<sub>B</sub> <math>f_{DCDCSYNCO} = f_{DCDC}</math>. The actual duty cycle is routed.      001<sub>B</sub> <math>f_{DCDCSYNCO} = f_{DCDC}/2</math>. Duty cycle is constant at 50%.      010<sub>B</sub> <math>f_{DCDCSYNCO} = f_{DCDC}/4</math>. Duty cycle is constant at 50%.      011<sub>B</sub> <math>f_{DCDCSYNCO} = f_{DCDC}/8</math>. Duty cycle is constant at 50%.      100<sub>B</sub> <math>f_{DCDCSYNCO} = f_{DCDC}/16</math>. Duty cycle is constant at 50%.      101<sub>B</sub> <math>f_{DCDCSYNCO} = f_{DCDC}/32</math>. Duty cycle is constant at 50%.</p>																																
<b>LCK</b>	31	rh	<p><b>Lock Status</b></p> <p>This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <p>0<sub>B</sub> The register is unlocked and can be updated      1<sub>B</sub> The register is locked and cannot be updated</p>																																
<b>0</b>	15:10, 30:27	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>																																

## Power Management System (PMS)

**Table 352 Access Mode Restrictions of EVRSDCTRL7 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	DRVNI, DRVP, DRVPCBF, DRVSLOMODE, DRVSPR, SYNCDIVFAC	
(default)	r	DRVNI, DRVP, DRVPCBF, DRVSLOMODE, DRVSPR, SYNCDIVFAC	

**Table 353 Reset Values of EVRSDCTRL7**

Reset Type	Reset Value	Note
LVD Reset	8000 00FE <sub>H</sub>	
Cold PORST	8000 00FE <sub>H</sub>	
After SSW execution	8000 00FE <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVRC SD Coefficient Register 6

#### EVRSDCOEFF6

#### EVRC SD Coefficient Register 6

(0160<sub>H</sub>)

Reset Value: [Table 355](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>															<b>CT5REG2</b>
rh				r											rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															<b>CT5REG1</b>
															<b>CT5REG0</b>
															rw

Field	Bits	Type	Description
<b>CT5REG0</b>	7:0	rw	<b>Commutation trimming and Slope Control(drv5v0_trim_i)</b> Trimming of the commutation parameters of the external driver (5V).
<b>CT5REG1</b>	15:8	rw	<b>Commutation trimming(drv5v1_trim_i)</b> Trimming of the commutation parameters of the external driver (5V).
<b>CT5REG2</b>	23:16	rw	<b>Commutation trimming(drv5v2_trim_i)</b> Trimming of the commutation parameters of the external driver (5V).
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
<b>0</b>	30:24	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System (PMS)

**Table 354 Access Mode Restrictions of EVRSDCOEFF6 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	CT5REG0, CT5REG1, CT5REG2	
(default)	r	CT5REG0, CT5REG1, CT5REG2	

**Table 355 Reset Values of EVRSDCOEFF6**

Reset Type	Reset Value	Note
LVD Reset	8097 1802 <sub>H</sub>	
Cold PORST	8097 1802 <sub>H</sub>	
After SSW execution	8097 1802 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVRC SD Coefficient Register 7

**EVRSDCOEFF7**

(0164 <sub>H</sub> )																Reset Value: Table 357			
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16																			
<b>LCK</b>																<b>0</b>			
rh																r			
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0																CT5REG4			
																CT5REG3			
																rw			

Field	Bits	Type	Description
CT5REG3	7:0	rw	<b>Commutation trimming(drv5v3_trim_i)</b> Trimming of the commutation parameters of the external driver (5V).
CT5REG4	15:8	rw	<b>Commutation trimming(drv5v4_trim_i)</b> Trimming of the commutation parameters of the external driver (5V).
LCK	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. $0_B$ The register is unlocked and can be updated $1_B$ The register is locked and cannot be updated
0	30:16	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 356 Access Mode Restrictions of EVRSDCOEFF7 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	CT5REG3, CT5REG4	
(default)	r	CT5REG3, CT5REG4	

## Power Management System (PMS)

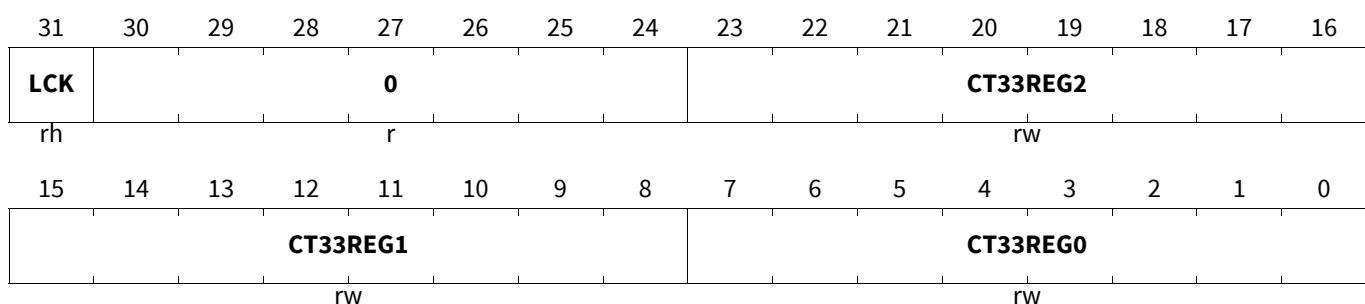
**Table 357 Reset Values of EVRSDCOEFF7**

Reset Type	Reset Value	Note
LVD Reset	8000 D8F7 <sub>H</sub>	
Cold PORST	8000 D8F7 <sub>H</sub>	
After SSW execution	8000 D8F7 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVRC SD Coefficient Register 8

**EVRSDCOEFF8**

**EVRC SD Coefficient Register 8 (0168<sub>H</sub>) Reset Value: Table 359**



Field	Bits	Type	Description
<b>CT33REG0</b>	7:0	rw	<b>Commutation trimming(drv3v0_trim_i)</b> Trimming of the commutation parameters of the external driver (3.3V).
<b>CT33REG1</b>	15:8	rw	<b>Commutation trimming(drv3v1_trim_i)</b> Trimming of the commutation parameters of the external driver (3.3V).
<b>CT33REG2</b>	23:16	rw	<b>Commutation trimming(drv3v2_trim_i)</b> Trimming of the commutation parameters of the external driver (3.3V).
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. $0_B$ The register is unlocked and can be updated $1_B$ The register is locked and cannot be updated
<b>0</b>	30:24	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 358 Access Mode Restrictions of EVRSDCOEFF8 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	CT33REG0, CT33REG1, CT33REG2	
(default)	r	CT33REG0, CT33REG1, CT33REG2	

## Power Management System (PMS)

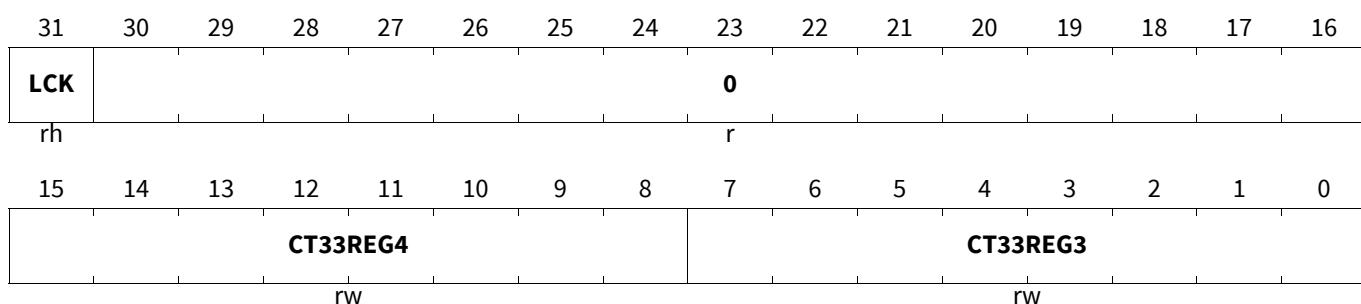
**Table 359 Reset Values of EVRSDCOEFF8**

Reset Type	Reset Value	Note
LVD Reset	8017 1002 <sub>H</sub>	
Cold PORST	8017 1002 <sub>H</sub>	
After SSW execution	8017 1002 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVRC SD Coefficient Register 9

**EVRSDCOEFF9**

**EVRC SD Coefficient Register 9 (016C<sub>H</sub>) Reset Value: Table 361**



Field	Bits	Type	Description
<b>CT33REG3</b>	7:0	rw	<b>Commutation trimming(drv3v3_trim_i)</b> Trimming of the commutation parameters of the external driver (3.3V).
<b>CT33REG4</b>	15:8	rw	<b>Commutation trimming(drv3v4_trim_i)</b> Trimming of the commutation parameters of the external driver (3.3V).
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. <b>0<sub>B</sub></b> The register is unlocked and can be updated <b>1<sub>B</sub></b> The register is locked and cannot be updated
<b>0</b>	30:16	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 360 Access Mode Restrictions of EVRSDCOEFF9 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	CT33REG3, CT33REG4	
(default)	r	CT33REG3, CT33REG4	

**Table 361 Reset Values of EVRSDCOEFF9**

Reset Type	Reset Value	Note
LVD Reset	8000 A0AF <sub>H</sub>	

## Power Management System (PMS)

**Table 361 Reset Values of EVRSDCOEFF9 (cont'd)**

Reset Type	Reset Value	Note
Cold PORST	8000 A0AF <sub>H</sub>	
After SSW execution	8000 A0AF <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVRC SD Control Register 8

#### EVRSDCTRL8

#### EVRC SD Control Register 8

(0128<sub>H</sub>)

Reset Value: [Table 363](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	0	FBADC LSB	0	FBADCERR	0	FBADCLPF	0	FBADCBLNK							
rh	r	rw	r	rw	r	rw	r	rw	r	rw	r	r	rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		FBADCSMP										FBADCOFFS			
r		rw										rw			

Field	Bits	Type	Description
<b>FBADCOFFS</b>	7:0	rw	<b>Feedback Converted Counter Value Offset(fbadc2_offset_i)</b> This bitfield configures the offset of the converted counter value of the feedback ADC measuring the core voltage.
<b>FBADCSMP</b>	13:8	rw	<b>FB ADC Sampling period(fbadc1_smpthr_i)</b> This bitfield configures the sampling period in 100 MHz clock cycles for the feedback ADC measuring the core voltage.
<b>FBADCBLNK</b>	17:16	rw	<b>FB ADC Blanked Samples Number(fbadc0_blank_i)</b> This bitfield configures the number of feedback ADC samples that are blanked in case of a transition of the PWM drive output to minimise switching noise influence.
<b>FBADCLPF</b>	21:20	rw	<b>FB ADC Counter LPF Coefficient(fbadc0_lpfcnt_i)</b> This bit field configures the coefficient of the Low Pass Filter of the feedback ADC counter value measuring the core voltage. $y [k] = \{y [k-1] * (1-a)\} + \{x [k] * a\}$ ; $y [k]$ is filter output; $x [k]$ is ADC output $a = \{1 / (2 ^ LPF)\}$ . If LPF = 0, the filter output is the same as ADC output.
<b>FBADCERR</b>	25:24	rw	<b>FB ADC Error LPF Coefficient(fbadc3_lpfeerr_i)</b> This bitfield configures the coefficient of the Low Pass Filter of the output voltage error signal of the feedback ADC.
<b>FBADCLSB</b>	28	rw	<b>FB ADC LSB for Error Computation(fbadc3_lsb_i)</b> This bitfield configures the LSB of the feedback ADC counter value used for the error computation. $O_B = 5 \text{ mV}$ $1_B = 10 \text{ mV}$

## Power Management System (PMS)

Field	Bits	Type	Description
LCK	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
0	15:14, 19:18, 23:22, 27:26, 30:29	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 362 Access Mode Restrictions of EVRSDCTRL8 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	FBADCBLNK, FBADCERR, FBADCLPF, FBADCLSB, FBADCOFFS, FBADCSMP	
(default)	r	FBADCBLNK, FBADCERR, FBADCLPF, FBADCLSB, FBADCOFFS, FBADCSMP	

**Table 363 Reset Values of EVRSDCTRL8**

Reset Type	Reset Value	Note
LVD Reset	9121 048E <sub>H</sub>	
Cold PORST	9121 048E <sub>H</sub>	
After SSW execution	9121 048E <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## EVRC SD Control Register 9

### EVRSDCTRL9

#### EVRC SD Control Register 9

(012C<sub>H</sub>)

Reset Value: [Table 365](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>								<b>0</b>							
rh								r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				<b>0</b>			<b>FFADCLPF</b>						<b>FFADCOFFS</b>		
				r			rw						rw		

Field	Bits	Type	Description
<b>FFADCOFFS</b>	7:0	rw	<b>Feed Forward Converted Counter Value Offset(ffadc1_offset_i)</b> This bit field configures the offset of the converted counter value of the feed forward ADC measuring the input VEXT voltage.

## Power Management System (PMS)

Field	Bits	Type	Description
FFADCLPF	10:8	rw	<b>FF ADC Counter LPF Coefficient(ffdadc0_lpcnt_i)</b> This bit field configures the coefficient of the Low Pass Filter of the feed-forward ADC counter value measuring the input VEXT voltage. $y[k] = \{y[k-1] * (1-a)\} + \{x[k] * a\}$ ; $y[k]$ is filter output; $x[k]$ is ADC output $a = \{1 / (2^LPF)\}$ . If LPF = 0, the filter output is the same as ADC output.
LCK	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. $0_B$ The register is unlocked and can be updated $1_B$ The register is locked and cannot be updated
0	30:11	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 364 Access Mode Restrictions of EVRSDCTRL9 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	FFADCLPF, FFADCOFFS	
(default)	r	FFADCLPF, FFADCOFFS	

**Table 365 Reset Values of EVRSDCTRL9**

Reset Type	Reset Value	Note
LVD Reset	8000 0434 <sub>H</sub>	
Cold PORST	8000 0434 <sub>H</sub>	
After SSW execution	8000 0434 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## EVRC SD Control Register 10

### EVRSDCTRL10

**EVRC SD Control Register 10** (0130<sub>H</sub>) Reset Value: [Table 366](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>0</b>	<b>SHLVE N</b>	<b>SHHV EN</b>							<b>0</b>						
r	rw	rw							r						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>SHVL</b>								<b>SHVH</b>							

Field	Bits	Type	Description
SHVH	7:0	rw	<b>Short to High Voltage Threshold(shrth1_shvh_i)</b> High Voltage Threshold = (SDVOUTSEL + SHVH x 5 mV). EVRC short to supply alarm has the nominal values of SHVH of 1.9V and tCSHHV of 3ms.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>SHVL</b>	15:8	rw	<b>Short to Low Voltage Threshold(shrtl1_shvl_i)</b> Low Voltage Threshold = (SDVOUTSEL - SHVL x 5 mV). EVRC short to ground alarm has the nominal values of SHVL of 0.8V and tCSHLV of 3ms.
<b>SHHVEN</b>	28	rw	<b>Short to High Detection Enable(shrth0_shhven_i)</b> $0_B$ Short to High Detection is disabled $1_B$ Short to High Detection is enabled
<b>SHLVEN</b>	29	rw	<b>Short to Low Detection Enable(shrtl0_shlven_i)</b> $0_B$ Short to Low Detection is disabled $1_B$ Short to Low Detection is enabled
<b>0</b>	27:16, 31:30	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 366 Reset Values of EVRSDCTRL10**

Reset Type	Reset Value	Note
LVD Reset	0000 5A82 <sub>H</sub>	
Cold PORST	0000 5A82 <sub>H</sub>	
After SSW execution	0000 5A82 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## EVRC SD Control Register 11

### EVRSDCTRL11

#### EVRC SD Control Register 11

(0134<sub>H</sub>)

Reset Value: [Table 368](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	<b>0</b>	<b>SYNCMUXSEL</b>	<b>0</b>		<b>SYNCHYST</b>			<b>0</b>					<b>SYNCMAXDEV</b>		
rh	r	rw		r	rw			r					rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		<b>0</b>			<b>DROOPVL</b>			<b>0</b>					<b>DROOPVH</b>		
		r			rw			r					rw		

Field	Bits	Type	Description
<b>DROOPVH</b>	4:0	rw	<b>High VDD Limit for Droop request(droopvh_thres_i)</b> This bitfield defines the VDD high voltage limit above which a positive droop request on VDD voltage shall be ignored. VDD Droop High Limit = 712.5 mV + LSB * (SDVOUTSEL+ SDVOUTTRIM+ DROOPVH); LSB = 5 mV
<b>DROOPVL</b>	12:8	rw	<b>Low VDD Limit for Droop request(droopvl_thres_i)</b> This bitfield defines the VDD low voltage limit below which a negative droop request on VDD voltage shall be ignored. VDD Droop Low Limit = 712.5 mV + LSB * (SDVOUTSEL+ SDVOUTTRIM- DROOPVL); LSB = 5 mV

## Power Management System (PMS)

Field	Bits	Type	Description
<b>SYNCMAXDEV</b>	20:16	rw	<p><b>Maximum Deviation of the Synchronization Input Frequency(synci1_maxdev_i)</b></p> <p>This bitfield defines the maximum allowed frequency deviation of the synchronization input signal frequency from the programmed nominal DCDC switching frequency (EVRSCTRL0.SDFREQ). For locking, EVRSCTRL11.SYNCMAXDEV has to be chosen to be greater or equal to the value of EVRSCTRL11.SYNCHYST, and unequal to zero. Violation of limit leads to loss of synchronization. The frequency window is defined as follows</p> $d f_{\text{MAXDEV}} = 100 \text{ MHz} * (2 * \text{SYNCMAXDEV}) / (\text{SDFREQ}^2 + \text{SYNCMAXDEV}^2)$ $\text{SYNCMAXDEV} = \text{round} [ (100 \text{ MHz} / d f_{\text{MAXDEV}}) - \sqrt{(100 \text{ MHz} / d f_{\text{MAXDEV}})^2 - \text{SDFREQ}^2} ]$
<b>SYNCHYST</b>	26:24	rw	<p><b>Lock Unlock Hysteresis Window(synci0_hyst_i)</b></p> <p>This bitfield defines the hysteresis window for synchronization locking and unlocking. For locking, EVRSCTRL11.SYNCHYST has to be chosen to be lower or equal to the value of EVRSCTRL11.SYNCMAXDEV, and unequal to zero. The limit is applied to the period counter running at 100 MHz.</p> <p>Upper unlock condition= SDFREQ + SYNCMAXDEV      Upper lock condition= SDFREQ + SYNCMAXDEV - SYNCHYST      Lower unlock condition = SDFREQ - SYNCMAXDEV      Lower lock condition = SDFREQ - SYNCMAXDEV + SYNCHYST  <math display="block">\text{SYNCHYST} = \text{round} [ d f_{\text{HYST}} * (SDFREQ \pm SYNCMAXDEV)^2 ] / [ d f_{\text{HYST}} * (SDFREQ \pm SYNCMAXDEV) + 100 \text{ MHz} ]</math> </p>
<b>SYNCMUXSEL</b>	29:28	rw	<p><b>Synchronisation Input Multiplexer</b></p> <p>This bitfield selects synchronisation input either from CCU6 or GTM inputs to be forwarded to EVRC SMPS regulator.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Synchronization input open or unconnected.</li> <li>01<sub>B</sub> CCU60 COUT63</li> <li>10<sub>B</sub> GTM</li> <li>11<sub>B</sub> Reserved</li> </ul>
<b>LCK</b>	31	rh	<p><b>Lock Status</b></p> <p>This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> The register is unlocked and can be updated</li> <li>1<sub>B</sub> The register is locked and cannot be updated</li> </ul>
<b>0</b>	7:5, 15:13, 23:21, 27, 30	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

## Power Management System (PMS)

**Table 367 Access Mode Restrictions of EVRSDCTRL11 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	DROOPVH, DROOPVL, SYNCHYST, SYNCMAXDEV, SYNCMUXSEL	
(default)	r	DROOPVH, DROOPVL, SYNCHYST, SYNCMAXDEV, SYNCMUXSEL	

**Table 368 Reset Values of EVRSDCTRL11**

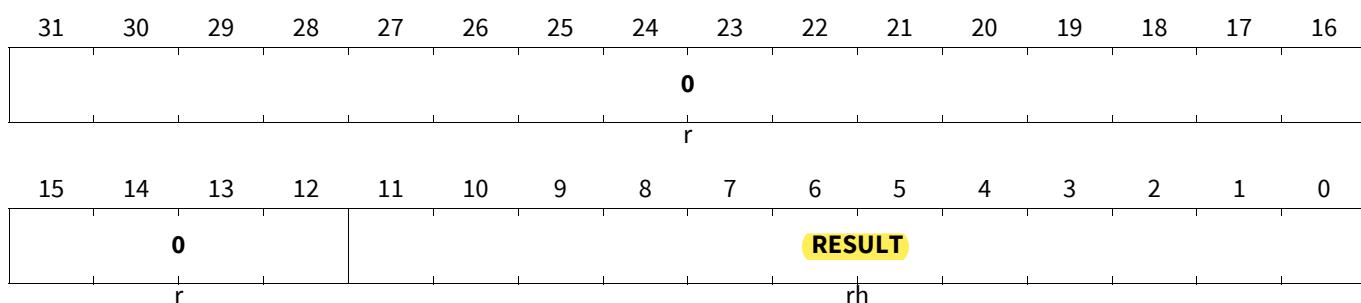
Reset Type	Reset Value	Note
LVD Reset	9207 0909 <sub>H</sub>	
Cold PORST	9207 0909 <sub>H</sub>	
After SSW execution	9207 0909 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### 11.3.1.3 Die Temperature Sensor Registers

#### Die Temperature Sensor Status Register

##### DTSSTAT

Die Temperature Sensor Status Register **(01C0<sub>H</sub>)** **Reset Value: Table 369**



Field	Bits	Type	Description
RESULT	11:0	rh	<b>Result of the DTS Measurement</b> This bit field shows the result of the DTS measurement. The value given is directly related to the die temperature and can be evaluated using the following formula. $T (\text{°C}) = [\text{RESULT} / \text{G}_{\text{nom}}] - 273.15$ $T (\text{°K}) = [\text{RESULT}] / \text{G}_{\text{nom}}$ $\text{RESULT} = \text{G}_{\text{nom}} * \{T (\text{°C}) + 273.15\} = \text{G}_{\text{nom}} * T (\text{°K})$ $\text{G}_{\text{nom}} = 7.505$
0	31:12	r	<b>Reserved</b> Read as 0.

## Power Management System (PMS)

**Table 369 Reset Values of DTSSTAT**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

### Die Temperature Sensor Limit Register

#### DTSLIM

Die Temperature Sensor Limit Register (01C8<sub>H</sub>) Reset Value: [Table 371](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>UOF</b>	<b>SLCK</b>	<b>0</b>													
rwh	rw	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>LLU</b>		<b>0</b>													
rwh		r													

Field	Bits	Type	Description
<b>LOWER</b>	11:0	rw	<b>Lower Limit</b> This bit field defines the lower limit of the DTS temperature check. The DTS measurement result is compared against this value and if the measurement result is less than or equal to the configured LOWER bitfield value; flag LLU is set.
<b>LLU</b>	15	rwh	<b>Lower Limit Underflow</b> When this bit is set, a HSM temperature underflow trigger is generated. When this bit is set the related SMU DTS alarm trigger is generated. This bit has to be written with zero in order to clear it. Writing a one has no effect. This bit is set when a DTS measurement is finished and the result is below the lower limit (i.e. DTSLIM.LOWER). 0 <sub>B</sub> No temperature underflow was detected 1 <sub>B</sub> A temperature underflow was detected
<b>UPPER</b>	27:16	rw	<b>Upper Limit</b> This bit field defines the upper limit of the DTS temperature check. The DTS measurement result is compared against this value and if the measurement result is greater than or equal to the configured UPPER bitfield value; flag UOF is set.
<b>SLCK</b>	30	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011 <sub>B</sub> ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. 0 <sub>B</sub> No lock active 1 <sub>B</sub> Lock is active

## Power Management System (PMS)

Field	Bits	Type	Description
<b>UOF</b>	31	rwh	<p><b>Upper Limit Overflow</b></p> <p>When this bit is set, a HSM temperature overflow trigger is generated.</p> <p>When this bit is set, the related SMU DTS alarm trigger is generated.</p> <p>This bit has to be written with zero in order to clear it. Writing a one has no effect.</p> <p>This bit is set when a DTS measurement is finished and the result is exceeding the upper limit (i.e. DTSLIM.UPPER).</p> <p>0<sub>B</sub> No temperature overflow was detected 1<sub>B</sub> A temperature overflow was detected</p>
<b>0</b>	14:12, 29:28	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 370 Access Mode Restrictions of DTSLIM sorted by descending priority**

Mode Name	Access Mode		Description
<b>SLCK = 0</b>	rw	LOWER, UPPER	
	rwh	LLU, UOF	
(default)	r	LOWER, SLCK, UPPER	
	rh	LLU, UOF	

**Table 371 Reset Values of DTSLIM**

Reset Type	Reset Value	Note
LVD Reset	0CD8 06D6 <sub>H</sub>	
Cold PORST	0CD8 06D6 <sub>H</sub>	

## Power Management System (PMS)

### 11.3.1.4 Standby and Wake-up Control Registers

#### Standby and Wake-up Control Register 0

##### PMSWCRO

**Standby and Wake-up Control Register 0 (00B4<sub>H</sub>) LVD Reset Value: 0010 02D0<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WUTW KEN	PORS TWKE N	SCRW KEN	PWRW KEN	PINBW KEN	PINAW KEN	ESR1 WKEN	ESR0 WKEN		BLNKFIL		0		STBYRAMSEL		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	r		rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PINBEDCON	PINBD FEN	PINAEDCON	PINAD FEN	ESR1EDCON	ESR1D FEN	ESR0EDCON	ESR0D FEN	VDDST BYEN	VEXTS TBYEN		0				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw			r

Field	Bits	Type	Description
VEXTSTBYEN	2	rw	<b>Standby Entry on VEXT Supply ramp-down</b> This bit field enables Standby Entry on VEXT supply ramp-down. This is supported only in case Standby domain is supplied separately via VEVRSSB supply pin and VEXT rail is switched off during Standby. The voltage threshold for entry is configured in EVRUVMON register. Current configuration is reflected in PMSWSTAT2.VEXTSTBYEN register bit. 0 <sub>B</sub> Standby Entry on VEXT supply ramp-down is disabled. 1 <sub>B</sub> Standby Entry triggered on a VEXT Supply undervoltage event (SWDUV). Blanking filter active on Standby mode entry.
VDDSTBYEN	3	rw	<b>Standby Entry on VDD Supply ramp-down</b> This bit field enables Standby Entry on VDD supply ramp-down. This is supported only in case Standby domain is supplied separately via VEVRSSB supply pin and VDD rail is switched off during Standby. The voltage threshold for entry is configured in EVRUVMON register. Current configuration is reflected in PMSWSTAT2.VDDSTBYEN register bit. 0 <sub>B</sub> Standby Entry on VDD supply ramp-down is disabled. 1 <sub>B</sub> Standby Entry triggered on a VDD Supply undervoltage event (VDDUV). Blanking filter active on Standby mode entry.
ESRODFEN	4	rw	<b>ESRO Digital Filter Enable</b> This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values) is enabled during normal RUN mode, then pulses less than 30ns are suppressed and pulses longer than 100ns will always result in a trigger. If the back-up clock is disabled in Standby mode and filter is running on 70 KHz Standby clock, then pulses less than 5 us are suppressed and pulses longer than 50 us will always result in a trigger. 0 <sub>B</sub> The filter is bypassed 1 <sub>B</sub> The filter is used

## Power Management System (PMS)

Field	Bits	Type	Description
<b>ESR0EDCON</b>	6:5	rw	<p><b>ESR0 Edge Detection Control</b></p> <p>This bit field defines the edge of a ESR0 wake-up trigger</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No trigger is generated</li> <li>01<sub>B</sub> A trigger is generated upon a rising edge</li> <li>10<sub>B</sub> A trigger is generated upon a falling edge</li> <li>11<sub>B</sub> A trigger is generated upon a rising OR falling edge</li> </ul>
<b>ESR1DFEN</b>	7	rw	<p><b>ESR1 Digital Filter Enable</b></p> <p>This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values) is enabled during normal RUN mode, then pulses less than 30ns are suppressed and pulses longer than 100ns will always result in a trigger. If the back-up clock is disabled in Standby mode and filter is running on 70 KHz Standby clock, then pulses less than 5 us are suppressed and pulses longer than 50 us will always result in a trigger.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> The filter is bypassed</li> <li>1<sub>B</sub> The filter is used</li> </ul>
<b>ESR1EDCON</b>	9:8	rw	<p><b>ESR1 Edge Detection Control</b></p> <p>This bit field defines the edge of a ESR1 wake-up trigger</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No trigger is generated</li> <li>01<sub>B</sub> A trigger is generated upon a rising edge</li> <li>10<sub>B</sub> A trigger is generated upon a falling edge</li> <li>11<sub>B</sub> A trigger is generated upon a rising OR falling edge</li> </ul>
<b>PINADFEN</b>	10	rw	<p><b>PINA Digital Filter Enable</b></p> <p>This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values) is enabled during normal RUN mode, then pulses less than 40ns are suppressed and pulses longer than 100ns will always result in a trigger. If the back-up clock is disabled in Standby mode and filter is running on 70 KHz Standby clock, then pulses less than 5 us are suppressed and pulses longer than 50 us will always result in a trigger.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> The filter is bypassed</li> <li>1<sub>B</sub> The filter is used</li> </ul>
<b>PINAEDCON</b>	12:11	rw	<p><b>PINA Edge Detection Control</b></p> <p>This bit field defines the edge of a Pin A wake-up trigger</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No trigger is generated</li> <li>01<sub>B</sub> A trigger is generated upon a rising edge</li> <li>10<sub>B</sub> A trigger is generated upon a falling edge</li> <li>11<sub>B</sub> A trigger is generated upon a rising OR falling edge</li> </ul>
<b>PINBDFEN</b>	13	rw	<p><b>PINB Digital Filter Enable</b></p> <p>This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values) is enabled during normal RUN mode, then pulses less than 40ns are suppressed and pulses longer than 100ns will always result in a trigger. If the back-up clock is disabled in Standby mode and filter is running on 70 KHz Standby clock, then pulses less than 5 us are suppressed and pulses longer than 50 us will always result in a trigger.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> The filter is bypassed</li> <li>1<sub>B</sub> The filter is used</li> </ul>

## Power Management System (PMS)

Field	Bits	Type	Description																												
<b>PINBEDCON</b>	15:14	rw	<p><b>PINB Edge Detection Control</b></p> <p>This bit field defines the edge of a Pin B wake-up trigger</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No trigger is generated</li> <li>01<sub>B</sub> A trigger is generated upon a rising edge</li> <li>10<sub>B</sub> A trigger is generated upon a falling edge</li> <li>11<sub>B</sub> A trigger is generated upon a rising OR falling edge</li> </ul>																												
<b>STBYRAMSEL</b>	18:16	rw	<p><b>Standby RAM supply in Standby Mode</b></p> <p>This bit field configures the Standby RAM blocks to be kept supplied during Standby Mode from VDDPD supply rail. The current configuration is reflected in PMSWSTAT2.STBYRAM bitfield.</p> <p><i>Note:</i> All other bit combinations are reserved.</p> <ul style="list-style-type: none"> <li>000<sub>B</sub> Standby RAM is not supplied.</li> <li>001<sub>B</sub> Standby RAM (CPU0 dLMU RAM Lower Half) is supplied.</li> <li>010<sub>B</sub> Standby RAM (CPU0 dLMU RAM) is supplied.</li> <li>100<sub>B</sub> Standby RAM (CPU1 dLMU RAM) is supplied.</li> <li>111<sub>B</sub> Standby RAMs (CPU0 dLMU &amp; CPU1 dLMU RAM) are supplied.</li> </ul>																												
<b>BLNKFIL</b>	23:20	rw	<p><b>Blanking Filter delay for Wake-up</b></p> <p>This bitfield enables a nominal blanking filter delay time immediately after Standby entry only after which a valid wake-up event is recognized and reacted upon. The actual delay may vary +- 30% to this nominal value. Current configuration is reflected in PMSWSTAT2.BLNKFIL bitfield.</p> <p><i>Note:</i> All other bit combinations are reserved. Incase WUT is used as a wake-up source, the blanking filter should be configured for a period greater than 3x 70kHz clock cycles.</p> <table> <tbody> <tr><td>0<sub>H</sub></td><td>0 ms</td></tr> <tr><td>1<sub>H</sub></td><td>2,5 ms</td></tr> <tr><td>2<sub>H</sub></td><td>5 ms</td></tr> <tr><td>3<sub>H</sub></td><td>10 ms</td></tr> <tr><td>4<sub>H</sub></td><td>20 ms</td></tr> <tr><td>5<sub>H</sub></td><td>40 ms</td></tr> <tr><td>6<sub>H</sub></td><td>80 ms</td></tr> <tr><td>7<sub>H</sub></td><td>160 ms</td></tr> <tr><td>8<sub>H</sub></td><td>320 ms</td></tr> <tr><td>9<sub>H</sub></td><td>640 ms</td></tr> <tr><td>A<sub>H</sub></td><td>1280 ms</td></tr> <tr><td>B<sub>H</sub></td><td>2560 ms</td></tr> <tr><td>C<sub>H</sub></td><td>5120 ms</td></tr> <tr><td>D<sub>H</sub></td><td>10240 ms</td></tr> </tbody> </table>	0 <sub>H</sub>	0 ms	1 <sub>H</sub>	2,5 ms	2 <sub>H</sub>	5 ms	3 <sub>H</sub>	10 ms	4 <sub>H</sub>	20 ms	5 <sub>H</sub>	40 ms	6 <sub>H</sub>	80 ms	7 <sub>H</sub>	160 ms	8 <sub>H</sub>	320 ms	9 <sub>H</sub>	640 ms	A <sub>H</sub>	1280 ms	B <sub>H</sub>	2560 ms	C <sub>H</sub>	5120 ms	D <sub>H</sub>	10240 ms
0 <sub>H</sub>	0 ms																														
1 <sub>H</sub>	2,5 ms																														
2 <sub>H</sub>	5 ms																														
3 <sub>H</sub>	10 ms																														
4 <sub>H</sub>	20 ms																														
5 <sub>H</sub>	40 ms																														
6 <sub>H</sub>	80 ms																														
7 <sub>H</sub>	160 ms																														
8 <sub>H</sub>	320 ms																														
9 <sub>H</sub>	640 ms																														
A <sub>H</sub>	1280 ms																														
B <sub>H</sub>	2560 ms																														
C <sub>H</sub>	5120 ms																														
D <sub>H</sub>	10240 ms																														
<b>ESROWKEN</b>	24	rw	<p><b>ESR0 Wake-up enable from Standby</b></p> <p>This bit configures wake-up via ESR0 pin from STANDBY mode and current configuration is reflected in PMSWSTAT2.ESR0WKEN register bit.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> System wake-up via ESR0 pin is disabled.</li> <li>1<sub>B</sub> System wake-up is enabled via ESR0 pin.</li> </ul>																												

## Power Management System (PMS)

Field	Bits	Type	Description
<b>ESR1WKEN</b>	25	rw	<b>ESR1 Wake-up enable from Standby</b> This bit configures wake-up via ESR1 pin from STANDBY mode and current configuration is reflected in PMSWSTAT2.ESR1WKEN register bit. $0_B$ System wake-up via ESR1 pin is disabled. $1_B$ System wake-up is enabled via ESR1 pin.
<b>PINAWKEN</b>	26	rw	<b>Pin A Wake-up enable from Standby</b> This bit configures wake-up via PINA pin from STANDBY mode and current configuration is reflected in PMSWSTAT2.PINAWKEN register bit. $0_B$ System wake-up via Pin A is disabled. $1_B$ System wake-up is enabled via Pin A.
<b>PINBWKEN</b>	27	rw	<b>Pin B Wake-up enable from Standby</b> This bit configures wake-up via PINB pin from STANDBY mode and current configuration is reflected in PMSWSTAT2.PINBWKEN register bit. $0_B$ System wake-up via Pin B is disabled. $1_B$ System wake-up is enabled via Pin B.
<b>PWRWKEN</b>	28	rw	<b>Standby Wake-up Enable on VEXT Supply ramp-up</b> This bit field enables wake-up on VEXT supply ramp-up after blanking filter time has expired. This is supported only in case Standby domain is supplied separately via VEVRSB supply pin and VEXT rail is switched off during Standby. Current configuration is reflected in PMSWSTAT2.PWRWKEN register bit. $0_B$ Wake-up on VEXT supply ramp-down is disabled. Blanking filter configuration has no effect. $1_B$ Wake-up from standby on VEXT supply ramp-up is enabled after blanking filter time expiry.
<b>SCRWKEN</b>	29	rw	<b>Standby Controller Wake-up enable from Standby</b> This bit configures wake-up via SCR from STANDBY mode and current configuration is reflected in PMSWSTAT2.SCRWKEN register bit. $0_B$ System wake-up via 8 bit Standby Controller is disabled. $1_B$ System wake-up is enabled via 8 bit Standby Controller.
<b>PORSTWKEN</b>	30	rw	<b>PORST pin Wake-up enable from Standby</b> This bit configures wake-up via PORST pin from STANDBY mode and current configuration is reflected in PMSWSTAT2.PORSTWKEN register bit. $0_B$ System wake-up via PORST pin is disabled. $1_B$ System wake-up via PORST pin is enabled.
<b>WUTWKEN</b>	31	rw	<b>WUT Wake-up enable from Standby</b> This bit configures wake-up via WUT from STANDBY mode and current configuration is reflected in PMSWSTAT2.WUTWKEN register bit. $0_B$ System wake-up via Wake-up Timer is disabled. $1_B$ System wake-up is enabled via Wake-up Timer.
<b>0</b>	1:0, 19	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System (PMS)

### Standby and Wake-up Control Register 2

#### PMSWCR2

#### Standby and Wake-up Control Register 2

(00B8<sub>H</sub>)LVD Reset Value: 0400 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
					<b>RST</b>	<b>SMUR ST</b>	<b>TCINT REQ</b>								<b>TCINT</b>
r				rh	rh	rh	rwh				rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				<b>SCRRS T</b>	<b>SCRW DT</b>	<b>SCREC C</b>	<b>0</b>								<b>SCRINT</b>
r				rwh	rwh	rwh	r				rh				

Field	Bits	Type	Description
<b>SCRINT</b>	7:0	rh	<b>Data exchange from Standby Controller to PMS main domain.</b> This bit field allows fast data exchange from SCR to PMS/CPUx. The data maybe read by CPUx consequent to an interrupt from the SCR to decode the interrupt. Incase SCR is enabled, at the end of the SCR Firmware routine, a value of 80H is set in SCRINT register to indicate that SCR has finished executing the startup code.
<b>SCRECC</b>	9	rwh	<b>SCR RAM ECC error / reset flag</b>  Note: <i>The flag is set by SCR and cleared by explicit write to the register bit. The flag is not cleared by SCR. While the SCR is being reset triggered by SCR RAM ECC error, this flag is set and clearing the flag is not possible for that duration.</i>  0 <sub>B</sub> No ECC error / reset reported by SCR. 1 <sub>B</sub> ECC error / reset was detected in SCR RAM.
<b>SCRWDT</b>	10	rwh	<b>SCR Watchdog Timer error / reset flag</b>  Note: <i>The flag is set by SCR and cleared by explicit write to the register bit. The flag is not cleared by SCR. While the SCR is being reset triggered by SCR watchdog, this flag is set and clearing the flag is not possible for that duration.</i>  0 <sub>B</sub> No WDT error / reset reported by SCR. 1 <sub>B</sub> WDT timer error / reset reported by SCR.
<b>SCRRST</b>	11	rwh	<b>SCR Software reset flag</b>  Note: <i>The flag is set by SCR and cleared by explicit write to the register bit. The flag is not cleared by SCR. While the SCR is being reset triggered by SCR software, this flag is set and clearing the flag is not possible for that duration.</i>  0 <sub>B</sub> No reset occurred in SCR. 1 <sub>B</sub> A reset has occurred in SCR.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>TCINT</b>	23:16	rw	<b>Data exchange from PMS main domain to Standby Controller.</b> This bit field allows fast data exchange from PMS to SCR. The data may be read by SCR consequent to an interrupt request (TCINTREQ) from PMS/CPUx to SCR to decode the interrupt.
<b>TCINTREQ</b>	24	rwh	<b>SW Interrupt request from PMS to Standby Controller.</b> Setting this bit triggers an interrupt to the 8 bit Standby controller.
<b>SMURST</b>	25	rh	<b>SMU Reset indication flag</b> $0_B$ No reset was issued by SMU. $1_B$ SMU issued an application or system reset.
<b>RST</b>	26	rh	<b>Application or System Reset indication flag</b> $0_B$ No application or system reset occurred. $1_B$ An application or system reset has occurred.
<b>0</b>	8, 15:12, 31:27	r	<b>Reserved</b> Read as 0; should be written with 0.

## Standby and Wake-up Control Register 3

**PMSWCR3****Standby and Wake-up Control Register 3**(00C0<sub>H</sub>)LVD Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	<b>WUTM ODE</b>	<b>WUTDI V</b>	<b>BUSY</b>	<b>WUTE N</b>	0										
r	rw	rw	rh	rw	r										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>WUTREL</b>															
rw															

Field	Bits	Type	Description
<b>WUTREL</b>	23:0	rw	<b>WUT reload value.</b> The counter starts counting down from WUTREL value. The current value of counter is indicated in WUTCNT. On WUTCNT underflow, a reload WUTCNT = WUTREL takes place in auto reload mode.
<b>WUTEN</b>	27	rw	<b>WUT enable</b> This bit enables the Wake-up Timer. The status bit PMSWSTAT.WUTEN is set once Wake-up Timer is enabled. $0_B$ Wake-up timer (WUT) disable request $1_B$ Wake-up timer (WUT) enable request.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>BUSY</b>	28	rh	<b>Lock Status - LCK</b> This bit indicates that the register is busy owing to ongoing bus access. The register can be updated with a new value when BUSY bit is cleared. The register requires synchronization to the 70kHz clock domain on a register update. 0 <sub>B</sub> The register can be updated. 1 <sub>B</sub> The register update is ongoing. A write action may stall bus access for the time duration BUSY bit is set.
<b>WUTDIV</b>	29	rw	<b>WUT clock divider</b> A write to this register bitfield may trigger immediate update irrespective of the status of BUSY bit. 0 <sub>B</sub> Wake-up timer (WUT) clock = fSB = 70 KHz clock. 1 <sub>B</sub> Wake-up timer (WUT) clock = fSB (70 KHz) / 210.
<b>WUTMODE</b>	30	rw	<b>WUT mode selection</b> This bit configures the Wake-up Timer mode. The status bit PMSWSTAT.WUTMODE is respectively updated. A write to this register bitfield may trigger immediate update irrespective of the status of BUSY bit. 0 <sub>B</sub> Wake-up timer (WUT) auto reload mode selected 1 <sub>B</sub> Wake-up timer (WUT) auto stop mode selected.
<b>0</b>	26:24, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

### Standby WUT Counter Register

**PMSWUTCNT****Standby WUT Counter Register****(00DC<sub>H</sub>)****LVD Reset Value: 0000 0000<sub>H</sub>**

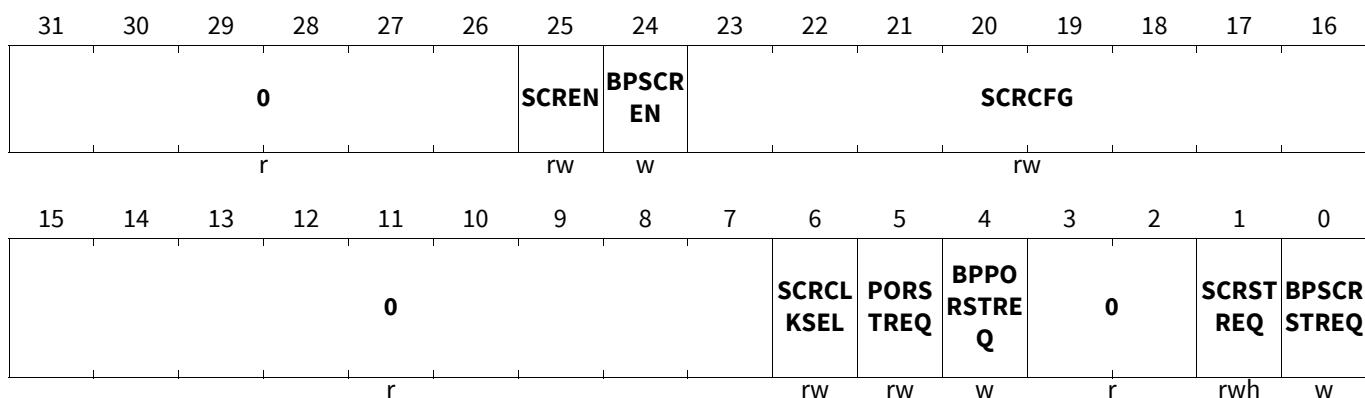
Field	Bits	Type	Description
<b>WUTCNT</b>	23:0	rh	<b>WUT counter value.</b> The current WUT counter value is indicated in this register bitfield. The WUTCNT value may have a deviation of 3 additional clock cycles to the expected counter value owing to synchronization overheads. The WUT clock is based on standby 70 kHz clock with ~ + - 30% variation. The counter depending on the mode can run through a RUN to STANDBY to RUN mode transition without interruption.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System (PMS)

### Standby and Wake-up Control Register 4

#### PMSWCR4

#### Standby and Wake-up Control Register 4

(00C4<sub>H</sub>)Reset Value: [Table 373](#)

Field	Bits	Type	Description
<b>BPSCRSTREQ</b>	0	w	<b>Standby Controller Reset request enable - SCRSTEN</b> $0_B$ Bit SCRSTREQ is not updated $1_B$ Bit SCRSTREQ can be updated
<b>SCRSTREQ</b>	1	rwh	<b>Standby Controller Reset request</b> $0_B$ No request for main reset of the 8 bit Standby Controller. $(evr\_scr\_rst\_req\_i)$ $1_B$ 8 bit Standby Controller reset request.
<b>BPPORSTREQ</b>	4	w	<b>Bit Protection for PORSTREQ - PORSTEN</b> $0_B$ Bit PORSTREQ is not updated $1_B$ Bit PORSTREQ can be updated
<b>PORSTREQ</b>	5	rw	<b>SCR Reset behavior on warm PORST in Normal RUN / SLEEP mode</b> $0_B$ 8 bit Standby Controller is not reset when warm PORST pin is asserted. $1_B$ 8 bit Standby Controller is reset when warm PORST pin is asserted. warm PORST usage in normal and standby mode.
<b>SCRCLKSEL</b>	6	rw	<b>Default Clock selection on Standby Mode Entry</b> $0_B$ 100MHz oscillator can be enabled or disabled based on request from Standby Controller. By default 100 MHz Oscillator is requested by SCR in Standby Mode. $1_B$ 100MHz oscillator is always active irrespective of SCR requests. Thus both 70 KHz Oscillator and 100 MHz oscillator are active in Standby Mode.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>SCRCFG</b>	23:16	rw	<p><b>Hardware configuration of the 8 bit SCR controller.</b></p> <p><b>Note:</b> Any change in SCRCFG is followed by a SCRSTREQ reset request of the 8 bit controller to start off in the chosen mode. All other bit combinations are reserved. Writing to PMSWCR4.SCRCFG with values != USERMODE1/0 will have an immediate effect on the enabling of debug pins.</p> <p>00<sub>H</sub> 8 bit XRAM is not programmed (default)      01<sub>H</sub> User Mode (Execution from 0000<sub>H</sub> XRAM address)      02<sub>H</sub> OCDS Mode (SCR DAP0_0/DAP1_0 pin mode)      03<sub>H</sub> OCDS Mode (SCR DAP0_1/DAP1_1 pin mode)      04<sub>H</sub> OCDS Mode (SCR SPD_0 pin mode)      05<sub>H</sub> OCDS Mode (SCR SPD_0 pin mode)      06<sub>H</sub> OCDS Mode (SCR SPD_1 pin mode)      07<sub>H</sub> OCDS Mode (SCR SPD_1 pin mode)      0A<sub>H</sub> OCDS Mode (SOC DAP mode)      0B<sub>H</sub> OCDS Mode (SOC DAP mode)      0C<sub>H</sub> OCDS Mode (SOC SPD mode)      0F<sub>H</sub> OCDS Mode (SOC SPD mode)</p>
<b>BPSCREN</b>	24	w	<p><b>Standby Controller Reset request enable</b></p> <p>0<sub>B</sub> Bit SCREN is not updated      1<sub>B</sub> Bit SCREN can be updated</p>
<b>SCREN</b>	25	rw	<p><b>Standby Controller Enable request</b></p> <p>SCR MBIST maybe activated independent of this bit.</p> <p>0<sub>B</sub> 8 bit Standby Controller is disabled      1<sub>B</sub> 8 bit Standby Controller is enabled</p>
<b>0</b>	3:2, 15:7, 31:26	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 372 Access Mode Restrictions of PMSWCR4 sorted by descending priority**

Mode Name	Access Mode		Description
write 1 to <b>BPSCRSTREQ</b>	rwh	SCRSTREQ	
write 1 to <b>BPPORSTREQ</b>	rw	PORSTREQ	
write 1 to <b>BPSCREN</b>	rw	SCREN	
(default)	r	PORSTREQ, SCREN	
	rh	SCRSTREQ	

## Power Management System (PMS)

**Table 373 Reset Values of PMSWCR4**

Reset Type	Reset Value	Note
LVD Reset	0000 0020 <sub>H</sub>	
After SSW execution	0200 0020 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets. SCR is initialized/started upon cold power-on reset which is not identified as exit from stand-by mode.

### Standby and Wake-up Control Register 5

Additional PORST digital filter activated via PORSTDF bit provides additional spike filtering of at least tPORSTDF duration to provide enhanced immunity against spurious spikes. This is in addition to the inherent analog PORST filter delay of the PORST pad / pin as documented in the datasheet. After cold PORST this delay is by default inactive.

#### PMSWCR5

**Standby and Wake-up Control Register 5 (00C8<sub>H</sub>) LVD Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								DCDC SYNC 0	0	PORS TDF	0	ESR0 RIST	TRIST REQ	BPTRI STREQ	w
r								rw	r	rw	r	rw	rwh	w	

Field	Bits	Type	Description
<b>BPTRISTREQ</b>	0	w	<b>Bit protection for Tristate request bit (TRISTREQ)</b> Setting this bit enables that bit TRISTREQ can be changed by a write operation. 0 <sub>B</sub> TRISTREQ keeps the previous state and cannot be changed. 1 <sub>B</sub> TRISTREQ bit can be changed with a write operation.
<b>TRISTREQ</b>	1	rwh	<b>Tristate enable</b> This bit decides whether pads behave as inputs with weak pull-up or tristate on reset assertion/de-assertion or Standby- Wake-up transition. After supply ramp-up or LVD reset, TRISTREQ = ! HWCFG6. 0 <sub>B</sub> No request to switch the input pad state of all the pads to tristate from pull-up (default reset state) 1 <sub>B</sub> Pad domain in tristate. VGATE1P pull up remains active if VEXT available and EVRC SMPS selected.
<b>ESR0TRIST</b>	2	rw	<b>ESR0 Tristate enable</b> This bit configures ESR0 pin behavior either as reset output or tristate during Standby mode if VEXT is supplied. 0 <sub>B</sub> ESR0 configured as reset output and is held low during Standby state (default reset state) 1 <sub>B</sub> ESR0 in tristate during Standby state.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>PORSTDF</b>	4	rw	<p><b>PORST Digital Filter enable</b></p> <p>This bit field enables additional PORST digital filter ( tPORSTDF parameter ) to provide enhanced immunity against spurious spikes.</p> <p>0<sub>B</sub> PORST recognition delay = Analog PORST pad filter delay (default reset state).</p> <p>1<sub>B</sub> PORST recognition delay = Analog PORST pad filter delay + Digital filter delay.</p>
<b>DCDCSYNCO</b>	6	rw	<p><b>DC-DC Synchronisation Output</b></p> <p>This bitfield enables the synchronisation output to synchronize the external SMPS regulator with respect to the internal EVRC regulator.</p> <p>0<sub>B</sub> DC-DC Synchronisation signal not available.</p> <p>1<sub>B</sub> DC-DC Synchronisation signal available.</p>
0	3, 5, 31:7	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 374 Access Mode Restrictions of PMSWCR5 sorted by descending priority**

Mode Name	Access Mode		Description
write 1 to <b>BPTRISTREQ</b>	rwh	TRISTREQ	
(default)	rh	TRISTREQ	

## Standby and Wake-up Status Register

### PMSWSTAT

#### Standby and Wake-up Status Register

(00D4<sub>H</sub>)

LVD Reset Value: 000A 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
<b>PINBI</b> NT	<b>PINAI</b> NT	<b>ESR1I</b> NT	<b>ESROI</b> NT	<b>0</b>	<b>WUTM</b> ODE	<b>WUTR</b> UN	<b>WUTE</b> N		<b>0</b>		<b>PORS</b> TREQ	<b>SCRCL</b> K	<b>SCRST</b>	<b>SCR</b>		
rh	rh	rh	rh	r	rh	rh	rh	rh	r	rh	rh	rh	rh	rh	rh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
					<b>PORS</b> TDF		<b>0</b>		<b>ESROT</b> RIST	<b>TESTM</b> ODE	<b>TRIST</b>	<b>HWCF</b> G5	<b>HWCF</b> G4	<b>0</b>	<b>HWCFGEV</b>	<b>0</b>
					rh	r	rh	rh	rh	rh	rh	r	rh	rh	r	

## Power Management System (PMS)

Field	Bits	Type	Description
<b>HWCFGEV</b>	2:1	rh	<p><b>EVR Hardware Configuration status</b></p> <p>This bit field indicates the supply configuration latched by the EVR from HWCFG[2:1] during a cold startup based on which EVRx regulators are consequently started. The latched configuration is used during STANDBY-RUN transition to reselect EVR mode.</p> <p>00<sub>B</sub> EVRC inactive, EVR33 inactive.      01<sub>B</sub> EVRC inactive, EVR33 active.      10<sub>B</sub> EVRC active, EVR33 inactive.      11<sub>B</sub> EVRC active, EVR33 active.</p>
<b>HWCFG4</b>	4	rh	<p><b>Hardware Configuration Pin 4 status</b></p> <p>This bit field indicates the latched level of HWCFG[4] during a cold startup.</p>
<b>HWCFG5</b>	5	rh	<p><b>Hardware Configuration Pin 5 status</b></p> <p>This bit field indicates the latched level of HWCFG[5] during a cold startup.</p>
<b>TRIST</b>	6	rh	<p><b>Pad Tristate / Pull-up status</b></p> <p>This bit indicates whether pads are configured as inputs with weak pull-up or as tristate during/after reset or after wake-up. At start-up, the value latched from HWCFG[6] pin decides the default state and is reflected in TRIST status bit. This bit may be later updated when PMSWCR5.TRISTREQ is set to override initial latched status from HWCFG[6].</p> <p>0<sub>B</sub> Pads configured as inputs with weak pull-up.      1<sub>B</sub> Pads are in tristate.</p>
<b>TESTMODE</b>	7	rh	<p><b>TESTMODE Pin status</b></p> <p>This bit field indicates the latched level of TESTMODE pin during a cold startup.</p>
<b>ESR0TRIST</b>	8	rh	<p><b>ESR0 pin status during Standby</b></p> <p>This bit indicates if ESR0 pin is configured as reset output or tristate during Standby mode &amp; transitions if VEXT is supplied. This bit is updated when PMSWCR5.ESR0TRIST is set.</p> <p>0<sub>B</sub> ESR0 configured as reset output and is held low during Standby state (default reset state)      1<sub>B</sub> ESR0 in tristate during Standby state.</p>
<b>PORSTDF</b>	11	rh	<p><b>PORST Digital Filter status</b></p> <p>This bit field indicates whether additional PORST digital filter is activated. This bit is updated when PMSWCR5.PORSTDF is set.</p> <p>0<sub>B</sub> PORST recognition delay = Analog PORST pad filter delay (default reset state).      1<sub>B</sub> PORST recognition delay = Analog PORST pad filter delay + Digital filter delay.</p>

## Power Management System (PMS)

Field	Bits	Type	Description
<b>SCR</b>	16	rh	<p><b>Standby Controller status</b></p> <p>This bit indicates whether SCR is enabled. This bit is updated when PMSWCR4.SCREN bit is set.</p> <p>0<sub>B</sub> 8 bit Standby Controller is disabled 1<sub>B</sub> 8 bit Standby Controller is enabled</p>
<b>SCRST</b>	17	rh	<p><b>Standby Controller Reset Indication flag</b></p> <p>This bit is set after a power-on reset as SCR is in reset state. This bit is consequently set when a reset is issued via PMSWCR4.SCRSTREQ bit. This status flag is set on every SCR reset caused by any SCR reset source.</p> <p>0<sub>B</sub> No reset of Standby controller took place. 1<sub>B</sub> Reset of Standby controller took place. (evr_scr_rst_o)</p>
<b>SCRCLK</b>	18	rh	<p><b>Current Clock configuration for SCR before Standby Mode Entry</b></p> <p>This bit is updated when PMSWCR4.SCRCLKSEL bit is set.</p> <p>0<sub>B</sub> Only 70 KHz Oscillator is active in Standby Mode. 1<sub>B</sub> Both 70 KHz Oscillator and 100 MHz oscillator are active in Standby Mode.</p>
<b>PORSTREQ</b>	19	rh	<p><b>Standby Controller Reset on warm PORST</b></p> <p>This bit is updated when PMSWCR4.PORSTREQ bit is set.</p> <p>0<sub>B</sub> 8 bit Standby Controller clock is not reset when warm PORST pin is asserted. 1<sub>B</sub> 8 bit Standby Controller is reset when warm PORST pin is asserted.</p>
<b>WUTEN</b>	24	rh	<p><b>WUT Enable status</b></p> <p>This bit indicates whether WUT is enabled. This bit is updated when PMSWCR3.WUTEN bit is updated.</p> <p>0<sub>B</sub> Wake-up timer (WUT) is disabled. 1<sub>B</sub> Wake-up timer (WUT) is enabled.</p>
<b>WUTRUN</b>	25	rh	<p><b>WUT Run status</b></p> <p>This bit indicates whether WUT is currently running. Due to synchronization to 70 KHz ( fSB ) WUT clock, setting of flag after enable may take up to 55 us.</p> <p>0<sub>B</sub> Wake-up timer (WUT) is inactive. 1<sub>B</sub> Wake-up timer (WUT) is active.</p>
<b>WUTMODE</b>	26	rh	<p><b>WUT Mode status</b></p> <p>This bit indicates the current WUT mode. This bit is updated when PMSWCR3.WUTMODE bit is updated.</p> <p>0<sub>B</sub> Wake-up timer (WUT) auto reload mode is selected 1<sub>B</sub> Wake-up timer (WUT) auto stop mode is selected.</p>
<b>ESR0INT</b>	28	rh	<p><b>ESR0 Interrupt flag</b></p> <p>In case interrupt was triggered by ESR0 pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via PMSWSTATCLR.ESR0INTCLR bit after interrupt is serviced.</p> <p>0<sub>B</sub> No interrupt event detected on ESR0 input. 1<sub>B</sub> An interrupt event as defined by PMSWCR0. ESR0EDCON detected on ESR0 input.</p>

## Power Management System (PMS)

Field	Bits	Type	Description
<b>ESR1INT</b>	29	rh	<b>ESR1 Interrupt flag</b> In case interrupt was triggered by ESR1 pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via PMSWSTATCLR.ESR1INTCLR bit after interrupt is serviced. 0 <sub>B</sub> No interrupt event detected on ESR1 input. 1 <sub>B</sub> An interrupt event as defined by PMSWCR0. ESR1EDCON detected on ESR1 input.
<b>PINAINT</b>	30	rh	<b>Pin A Interrupt flag</b> In case interrupt was triggered by PINA pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via PMSWSTATCLR.PINAINTCLR bit after interrupt is serviced. 0 <sub>B</sub> No interrupt event detected on Pin A input. 1 <sub>B</sub> An interrupt event as defined by PMSWCR0. PINAEDCON detected on Pin A input.
<b>PINBINT</b>	31	rh	<b>Pin B Interrupt flag</b> In case interrupt was triggered by PINB pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via PMSWSTATCLR.PINBINTCLR bit after interrupt is serviced. 0 <sub>B</sub> No interrupt event detected on the Pin B input. 1 <sub>B</sub> An interrupt event as defined by PMSWCR0. PINBEDCON detected on Pin B input.
<b>0</b>	0, 3, 10:9, 15:12, 23:20, 27	r	<b>Reserved</b> Read as 0; should be written with 0.

## Standby and Wake-up Status Register 2

### PMSWSTAT2

#### Standby and Wake-up Status Register 2

(00D8<sub>H</sub>)

LVD Reset Value: 0010 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WUTW KEN	PORS TWKE N	SCRW KEN	PWRW KEN	PINBW KEN	PINAW KEN	ESR1 WKEN	ESR0 WKEN		BLNKFIL		VEXTS TBYEN		STBYRAM		
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WUTO VRUN	PORS TOVR UN	SCRO VRUN	VDDST BYEN	PINBO VRUN	PINAO VRUN	ESR10 VRUN	ESR00 VRUN	WUTW KP	PORS TWKP	SCRW KP	PWRW KP	PINBW KP	PINAW KP	ESR1 WKP	ESR0 WKP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

## Power Management System (PMS)

Field	Bits	Type	Description
<b>ESR0WKP</b>	0	rh	<b>ESR0 Wake-up flag</b> In case wake-up was triggered by ESR0 pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.ESR0WKPCR bit before next STANDBY entry. 0 <sub>B</sub> No wake-up event detected on ESR0 input during STANDBY. 1 <sub>B</sub> An event as defined by PMSWCR0. ESR0EDCON detected on ESR0 input.
<b>ESR1WKP</b>	1	rh	<b>ESR1 Wake-up flag</b> In case wake-up was triggered by ESR1 pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.ESR1WKPCR bit before next STANDBY entry. 0 <sub>B</sub> No wake-up event detected on ESR1 input during STANDBY. 1 <sub>B</sub> An event as defined by PMSWCR0. ESR1EDCON detected on ESR1 input.
<b>PINAWKP</b>	2	rh	<b>Pin Wake-up flag</b> In case wake-up was triggered by PINA pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PINAWKPCR bit before next STANDBY entry. 0 <sub>B</sub> No wake-up event detected on Pin A input during STANDBY. 1 <sub>B</sub> An event as defined by PMSWCR0. PINAEDCON detected on Pin A input.
<b>PINBWKP</b>	3	rh	<b>Pin B Wake-up flag</b> In case wake-up was triggered by PINB pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PINBKPCR bit before next STANDBY entry. 0 <sub>B</sub> No wake-up event occurred on the Pin B input during STANDBY. 1 <sub>B</sub> An event as defined by PMSWCR0. PINBEDCON detected on Pin B input.
<b>PWRWKP</b>	4	rh	<b>Wake-up event on VEXT Supply ramp-up</b> In case wake-up was triggered by VEXT ramp-up pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PWRWKPCR bit before next STANDBY entry. 0 <sub>B</sub> No VEXT supply wake-up event detected. 1 <sub>B</sub> VEXT Monitor threshold exceeded on VEXT supply ramp-up leading to System Wake-up from STANDBY.
<b>SCRWKP</b>	5	rh	<b>SCR Wake-up flag</b> In case wake-up is triggered by SCR to the main controller during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.SCRWKPCR bit before next STANDBY entry. 0 <sub>B</sub> No SCR wake-up event detected. 1 <sub>B</sub> A SCR wake-up event occurred.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>PORSTWKP</b>	6	rh	<b>PORST Wake-up flag</b> In case wake-up was triggered by PORST pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PORSTWKPCR bit before next STANDBY entry. 0 <sub>B</sub> No wake-up event detected on PORST input during STANDBY if enabled via PMSWCR0.PORSTWKEN bit. 1 <sub>B</sub> A wake-up event detected on PORST input if enabled via PMSWCR0.PORSTWKEN bit.
<b>WUTWKP</b>	7	rh	<b>WUT Wake-up flag</b> In case wake-up was triggered by Wake-up timer during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.WUTWKPCR bit before next STANDBY entry. 0 <sub>B</sub> No wake-up event detected due to WUT underflow. 1 <sub>B</sub> A wake-up event from STANDBY was detected due to WUT underflow.
<b>ESR0OVRUN</b>	8	rh	<b>ESR0 Overrun status flag</b> This flag indicates that a consecutive ESR0 wake-up event occurred while ESR0WKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.ESR0OVRUNCLR bit before next STANDBY entry. 0 <sub>B</sub> No overrun condition detected on ESR0 input. 1 <sub>B</sub> An overrun condition detected on ESR0 input.
<b>ESR1OVRUN</b>	9	rh	<b>ESR1 Overrun status flag</b> This flag indicates that a consecutive ESR1 wake-up event occurred while ESR1WKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.ESR1OVRUNCLR bit before next STANDBY entry. 0 <sub>B</sub> No overrun condition detected on ESR1 input. 1 <sub>B</sub> An overrun condition detected on ESR1 input.
<b>PINAOVRUN</b>	10	rh	<b>Pin A Overrun status flag</b> This flag indicates that a consecutive PINA wake-up event occurred while PINAWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PINAOVRUNCLR bit before next STANDBY entry. 0 <sub>B</sub> No overrun condition detected on Pin A input. 1 <sub>B</sub> An overrun condition detected on Pin A input.
<b>PINBOVRUN</b>	11	rh	<b>Pin B Overrun status flag</b> This flag indicates that a consecutive PINB wake-up event occurred while PINBWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PINBOVRUNCLR bit before next STANDBY entry. 0 <sub>B</sub> No overrun condition detected on Pin B input. 1 <sub>B</sub> An overrun condition detected on Pin B input.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>VDDSTBYEN</b>	12	rh	<p><b>Standby Entry Enable status on VDD Supply ramp-down - VDDSTBYWKEN</b></p> <p>This bit indicates that Standby Entry may be triggered on a VDD Supply undervoltage event (VDDUV). This is supported only when Standby domain is supplied separately by VEVRSB Standby supply pin. This bit is updated when PMSWCRO.VDDSTBYWKEN bit is updated.</p> <p>0<sub>B</sub> 0 Standby Entry on VDD supply ramp-down is disabled. 1<sub>B</sub> 1 Standby Entry is enabled on a VDD Supply undervoltage event (SWDUV). Blanking filter active on Standby mode entry.</p>
<b>SCROVRUN</b>	13	rh	<p><b>SCR Overrun status flag</b></p> <p>This flag indicates that a consecutive SCR wake-up event occurred while SCRWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.SCROVRUNCLR bit before next STANDBY entry.</p> <p>0<sub>B</sub> No overrun condition detected of SCR wake-up event. 1<sub>B</sub> An overrun condition detected of SCR wake-up event.</p>
<b>PORSTOVRUN</b>	14	rh	<p><b>PORST Overrun status flag</b></p> <p>This flag indicates that a consecutive PORST wake-up event occurred while PORSTWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PORSTOVRUNCLR bit before next STANDBY entry.</p> <p>0<sub>B</sub> No overrun condition detected on PORST input if enabled via PMSWCRO.PORSTWKEN bit. 1<sub>B</sub> An overrun condition detected on PORST input if enabled via PMSWCRO.PORSTWKEN bit.</p>
<b>WUTOVRUN</b>	15	rh	<p><b>WUT Overrun status flag</b></p> <p>This flag indicates that a consecutive WUT wake-up event occurred while WUTWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.WUTOVRUNCLR bit before next STANDBY entry. WUTREL need to be greater than 10 during Standby mode to be able to latch consecutive WUT underflow events and update the WUTOVRRUN register bitfield.</p> <p>0<sub>B</sub> No overrun condition detected of WUT events. 1<sub>B</sub> An overrun condition detected of WUT events.</p>

## Power Management System (PMS)

Field	Bits	Type	Description																												
<b>STBYRAM</b>	18:16	rh	<p><b>Standby RAM Supply status</b></p> <p>This bit field indicates whether Standby RAM was supplied during Standby Mode and to infer status after a wake-up event. This bit is updated when PMSWCR0.STBYRAMSEL is set.</p> <p><b>Note:</b> <i>All other bit combinations are reserved. In case of VDDPD Standby supply fail or VEVRSB supply fail leading to LVD reset (indicated also in RSTSTAT.STBYR), the STBYRAM status bit is reset to 000<sub>B</sub> to indicate that Standby RAM contents may be corrupted.</i></p> <p>000<sub>B</sub> Standby RAM is not supplied.      001<sub>B</sub> Standby RAM (CPU0 dLMU RAM Lower Half) is supplied.      010<sub>B</sub> Standby RAM (CPU0 dLMU RAM) is supplied.      100<sub>B</sub> Standby RAM (CPU1 dLMU RAM) is supplied.      111<sub>B</sub> Standby RAMs (CPU0 dLMU &amp; CPU1 dLMU) are supplied.</p>																												
<b>VEXTSTBYEN</b>	19	rh	<p><b>Standby Entry Enable status on VEXT Supply ramp-down - VEXTSTBYWKEN</b></p> <p>This bit indicates that Standby Entry may be triggered on a VEXT Supply undervoltage event (SWDUV). This is supported only when Standby domain is supplied separately by VEVRSB Standby supply pin. This bit is updated when PMSWCR0.VEXTSTBYWKEN bit is updated.</p> <p>0<sub>B</sub> 0 Standby Entry on VEXT supply ramp-down is disabled.      1<sub>B</sub> 1 Standby Entry is enabled on a VEXT Supply undervoltage event (SWDUV). Blanking filter active on Standby mode entry.</p>																												
<b>BLNKFIL</b>	23:20	rh	<p><b>Blanking Filter Delay for VEXT Supply Wake-up</b></p> <p>This bit field indicates the Blanking filter configuration. This bit field is updated with the value configured in PMSWCR0.BLNKFIL bitfield.</p> <p><b>Note:</b> <i>All other bit combinations are reserved.</i></p> <table> <tbody> <tr> <td>0<sub>H</sub></td> <td>0 ms</td> </tr> <tr> <td>1<sub>H</sub></td> <td>2,5 ms</td> </tr> <tr> <td>2<sub>H</sub></td> <td>5 ms</td> </tr> <tr> <td>3<sub>H</sub></td> <td>10 ms</td> </tr> <tr> <td>4<sub>H</sub></td> <td>20 ms</td> </tr> <tr> <td>5<sub>H</sub></td> <td>40 ms</td> </tr> <tr> <td>6<sub>H</sub></td> <td>80 ms</td> </tr> <tr> <td>7<sub>H</sub></td> <td>160 ms</td> </tr> <tr> <td>8<sub>H</sub></td> <td>320 ms</td> </tr> <tr> <td>9<sub>H</sub></td> <td>640 ms</td> </tr> <tr> <td>A<sub>H</sub></td> <td>1280 ms</td> </tr> <tr> <td>B<sub>H</sub></td> <td>2560 ms</td> </tr> <tr> <td>C<sub>H</sub></td> <td>5120 ms</td> </tr> <tr> <td>D<sub>H</sub></td> <td>10240 ms</td> </tr> </tbody> </table>	0 <sub>H</sub>	0 ms	1 <sub>H</sub>	2,5 ms	2 <sub>H</sub>	5 ms	3 <sub>H</sub>	10 ms	4 <sub>H</sub>	20 ms	5 <sub>H</sub>	40 ms	6 <sub>H</sub>	80 ms	7 <sub>H</sub>	160 ms	8 <sub>H</sub>	320 ms	9 <sub>H</sub>	640 ms	A <sub>H</sub>	1280 ms	B <sub>H</sub>	2560 ms	C <sub>H</sub>	5120 ms	D <sub>H</sub>	10240 ms
0 <sub>H</sub>	0 ms																														
1 <sub>H</sub>	2,5 ms																														
2 <sub>H</sub>	5 ms																														
3 <sub>H</sub>	10 ms																														
4 <sub>H</sub>	20 ms																														
5 <sub>H</sub>	40 ms																														
6 <sub>H</sub>	80 ms																														
7 <sub>H</sub>	160 ms																														
8 <sub>H</sub>	320 ms																														
9 <sub>H</sub>	640 ms																														
A <sub>H</sub>	1280 ms																														
B <sub>H</sub>	2560 ms																														
C <sub>H</sub>	5120 ms																														
D <sub>H</sub>	10240 ms																														

## Power Management System (PMS)

Field	Bits	Type	Description
<b>ESR0WKEN</b>	24	rh	<b>ESR0 Wake-up enable status</b> This bit indicates that ESR0 is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.ESR0WKEN bit is updated. $0_B$ Wake-up from Standby via ESR0 is disabled. $1_B$ Wake-up from Standby via ESR0 is enabled.
<b>ESR1WKEN</b>	25	rh	<b>ESR1 Wake-up enable status</b> This bit indicates that ESR1 is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.ESR1WKEN bit is updated. $0_B$ Wake-up from Standby via ESR1 is disabled. $1_B$ Wake-up from Standby via ESR1 is enabled.
<b>PINAWKEN</b>	26	rh	<b>Pin A Wake-up enable status</b> This bit indicates that PINA is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.PINAWKEN bit is updated. $0_B$ Wake-up from Standby via PINA is disabled. $1_B$ Wake-up from Standby via PINA is enabled.
<b>PINBWKEN</b>	27	rh	<b>Pin B Wake-up enable status</b> This bit indicates that PINB is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.PINBWKEN bit is updated. $0_B$ Wake-up from Standby via PINB is disabled. $1_B$ Wake-up from Standby via PINB is enabled.
<b>PWRWKEN</b>	28	rh	<b>Standby Wake-up Enable status on VEXT Supply ramp-up</b> This bit indicates that VEXT detector is enabled to trigger wake-up from Standby during VEXT supply ramp-up after blanking filter time has expired. This is supported only when Standby domain is supplied separately by VEVRSB Standby supply pin. This bit is updated when PMSWCR0.PWRWKEN bit is updated. $0_B$ Wake-up on VEXT supply ramp-down disabled. Blanking filter configuration has no effect. $1_B$ Standby Wake-up on VEXT supply ramp-up is enabled after blanking filter expiry.
<b>SCRWKEN</b>	29	rh	<b>Standby Controller Wake-up Enable status</b> This bit indicates that SCR is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.SCRWKEN bit is updated. $0_B$ Wake-up from Standby via SCR is disabled. $1_B$ Wake-up from Standby via SCR is enabled.
<b>PORSTWKEN</b>	30	rh	<b>PORST pin Wake-up enable status from Standby</b> This bit indicates that wake-up via PORST pin is enabled during STANDBY mode. This bit is updated when PMSWCR0. PORSTWKEN bit is updated. $0_B$ System wake-up via PORST pin is disabled. $1_B$ System wake-up via PORST pin is enabled.
<b>WUTWKEN</b>	31	rh	<b>WUT Wake-up enable status</b> This bit indicates that WUT is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.WUTWKEN bit is updated. $0_B$ Wake-up from Standby via WUT is disabled. $1_B$ Wake-up from Standby via WUT is enabled.

## Power Management System (PMS)

### Standby and Wake-up Status Clear Register

#### PMSWSTATCLR

**Standby and Wake-up Status Clear Register (00E8<sub>H</sub>)**

**LVD Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PINBI NTCLR	PINAI NTCLR	ESR1I NTCLR	ESROI NTCLR						0						SCRST CLR
W	W	W	W						r						W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WUTO VRUN CLR	PORS TOVR UNCL R	SCRO VRUN CLR	0	PINBO VRUN CLR	PINA0 VRUN CLR	ESR10 VRUN CLR	ESR00 VRUN CLR	WUTW KPCLR	PORS TWKP CLR	SCRW KPCLR	PWRW KPCLR	PINBW KPCLR	PINAW KPCLR	ESR1 WKPC LR	ESR0 WKPC LR
W	W	W	r	W	W	W	W	W	W	W	W	W	W	W	W

Field	Bits	Type	Description
ESR0WKPCLR	0	w	<b>ESR0 Wake-up indication flag clear</b> $0_B$ No action $1_B$ PMSWSTAT2.ESR0WKP bit cleared.
ESR1WKPCLR	1	w	<b>ESR1 Wake-up indication flag clear</b> $0_B$ No action $1_B$ PMSWSTAT2.ESR1WKP bit cleared.
PINAWKPCLR	2	w	<b>PINA Wake-up indication flag clear</b> $0_B$ No action $1_B$ PMSWSTAT2.PINAWKP bit cleared.
PINBWKPCLR	3	w	<b>PINB Wake-up indication flag clear</b> $0_B$ No action $1_B$ PMSWSTAT2.PINBWKP bit cleared.
PWRWKPCLR	4	w	<b>PWRWKP Wake-up indication flag clear</b> $0_B$ No action $1_B$ PMSWSTAT2.PWRWKP bit cleared.
SCRWKPCLR	5	w	<b>SCR Wake-up indication flag clear</b> $0_B$ No action $1_B$ PMSWSTAT2.SCRWKP bit cleared.
PORSTWKPCR	6	w	<b>PORST Wake-up indication flag clear</b> $0_B$ No action $1_B$ PMSWSTAT2.PORSTWKP bit cleared.
WUTWKPCLR	7	w	<b>WUT Wake-up indication flag clear</b> $0_B$ No action $1_B$ PMSWSTAT2.WUTWKP bit cleared.
ESR0OVRUNC LR	8	w	<b>ESR0 Overrun status indication flag clear</b> $0_B$ No action $1_B$ PMSWSTAT2.ESR0OVRUN bit cleared.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>ESR1OVRUNC</b> <b>LR</b>	9	w	<b>ESR1 Overrun status indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.ESR1OVRUN bit cleared.
<b>PINAOVRUNC</b> <b>LR</b>	10	w	<b>PINA Overrun status indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.PINAOVRUN bit cleared.
<b>PINBOVRUNC</b> <b>LR</b>	11	w	<b>PINB Overrun status indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.PINBOVRUN bit cleared.
<b>SCROVRUNCL</b> <b>R</b>	13	w	<b>SCR Overrun status indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.SCROVRUN bit cleared.
<b>PORSTOVRUN</b> <b>CLR</b>	14	w	<b>PORST Overrun status indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.PORSTOVRUN bit cleared.
<b>WUTOVRUNC</b> <b>LR</b>	15	w	<b>WUT Overrun status indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.WUTOVRUN bit cleared.
<b>SCRSTCLR</b>	16	w	<b>Standby controller SCRST indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT.SCRST bit cleared.
<b>ESR0INTCLR</b>	28	w	<b>ESR0 Interrupt indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT.ESR0INT bit cleared.
<b>ESR1INTCLR</b>	29	w	<b>ESR1 Interrupt indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT.ESR1INT bit cleared.
<b>PINAINTCLR</b>	30	w	<b>PINA Interrupt indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT.PINAINT bit cleared.
<b>PINBINTCLR</b>	31	w	<b>PINB Interrupt indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT.PINBINT bit cleared.
<b>0</b>	12, 27:17	r	<b>Reserved</b> Read as 0; should be written with 0.

## 11.3.1.5 OCDS Trigger Bus Configuration Registers (OTGB)

Access are only supported for byte, half-word and word data and requires Supervisor Mode.

## Power Management System (PMS)

### OCDS Trigger Set Select Register

#### OTSS

#### OCDS Trigger Set Select Register

(01E0<sub>H</sub>)Reset Value: [Table 375](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
r															
0				OTGB1				0				OTGB0			
r								r							

Field	Bits	Type	Description
OTGB0	3:0	rw	<b>Trigger Set for OTGB0</b> 0 <sub>H</sub> No Trigger Set selected 1 <sub>H</sub> Trigger Set TS16_ADCMON 2 <sub>H</sub> Trigger Set TS16_EVRC <b>others</b> , reserved
OTGB1	11:8	rw	<b>Trigger Set for OTGB1</b> 0 <sub>H</sub> No Trigger Set selected 1 <sub>H</sub> Trigger Set TS16_ADCMON 2 <sub>H</sub> Trigger Set TS16_EVRC <b>others</b> , reserved
0	7:4, 15:12, 31:16	r	<b>Reserved</b> Read as 0; must be written with 0.

**Table 375 Reset Values of OTSS**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Debug Reset	0000 0000 <sub>H</sub>	

### OCDS Trigger Set Control 0 Register

#### OTSC0

#### OCDS Trigger Set Control 0 Register

(01E4<sub>H</sub>)Reset Value: [Table 376](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r									r						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
r									r						
0				B1HAM				0				B1LAM			
0				B0HAM				0				B0LAM			

## Power Management System (PMS)

Field	Bits	Type	Description
<b>B0LAM</b>	3:0	rw	<b>OTGB0 TS16_ADCMON Low Byte</b> 0 <sub>H</sub> No Module selected 1 <sub>H</sub> PRADCCV 2 <sub>H</sub> PRADC33V 3 <sub>H</sub> PRADCSWDV 4 <sub>H</sub> PRADCFBCV 5 <sub>H</sub> SECADCCV 6 <sub>H</sub> SECADC33V 7 <sub>H</sub> SECADCSWDV 8 <sub>H</sub> SECADCPRE 9 <sub>H</sub> SECACDSB A <sub>H</sub> SECACVDDM B <sub>H</sub> DTSRESULTL C <sub>H</sub> DTSRESULTH <b>others</b> , reserved
<b>B0HAM</b>	11:8	rw	<b>OTGB0 TS16_ADCMON High Byte</b> 0 <sub>H</sub> No Module selected 1 <sub>H</sub> PRADCCV 2 <sub>H</sub> PRADC33V 3 <sub>H</sub> PRADCSWDV 4 <sub>H</sub> PRADCFBCV 5 <sub>H</sub> SECADCCV 6 <sub>H</sub> SECADC33V 7 <sub>H</sub> SECADCSWDV 8 <sub>H</sub> SECADCPRE 9 <sub>H</sub> SECACDSB A <sub>H</sub> SECACVDDM B <sub>H</sub> DTSRESULTL C <sub>H</sub> DTSRESULTH <b>others</b> , reserved
<b>B1LAM</b>	19:16	rw	<b>OTGB1 TS16_ADCMON Low Byte</b> 0 <sub>H</sub> No Module selected 1 <sub>H</sub> PRADCCV 2 <sub>H</sub> PRADC33V 3 <sub>H</sub> PRADCSWDV 4 <sub>H</sub> PRADCFBCV 5 <sub>H</sub> SECADCCV 6 <sub>H</sub> SECADC33V 7 <sub>H</sub> SECADCSWDV 8 <sub>H</sub> SECADCPRE 9 <sub>H</sub> SECACDSB A <sub>H</sub> SECACVDDM B <sub>H</sub> DTSRESULTL C <sub>H</sub> DTSRESULTH <b>others</b> , reserved

## Power Management System (PMS)

Field	Bits	Type	Description
<b>B1HAM</b>	27:24	rw	<b>OTGB1_TS16_ADCMON High Byte</b> 0 <sub>H</sub> No Module selected 1 <sub>H</sub> PRADCCV 2 <sub>H</sub> PRADC33V 3 <sub>H</sub> PRADCSDV 4 <sub>H</sub> PRADCFCV 5 <sub>H</sub> SECADCCV 6 <sub>H</sub> SECADC33V 7 <sub>H</sub> SECADCSDV 8 <sub>H</sub> SECADCPRE 9 <sub>H</sub> SECADCSB A <sub>H</sub> SECADCVDDM B <sub>H</sub> DTSRESULTL C <sub>H</sub> DTSRESULTH <b>others</b> , reserved
<b>0</b>	7:4, 15:12, 23:20, 31:28	r	<b>Reserved</b> Read as 0; must be written with 0.

**Table 376 Reset Values of OTSCO**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Debug Reset	0000 0000 <sub>H</sub>	

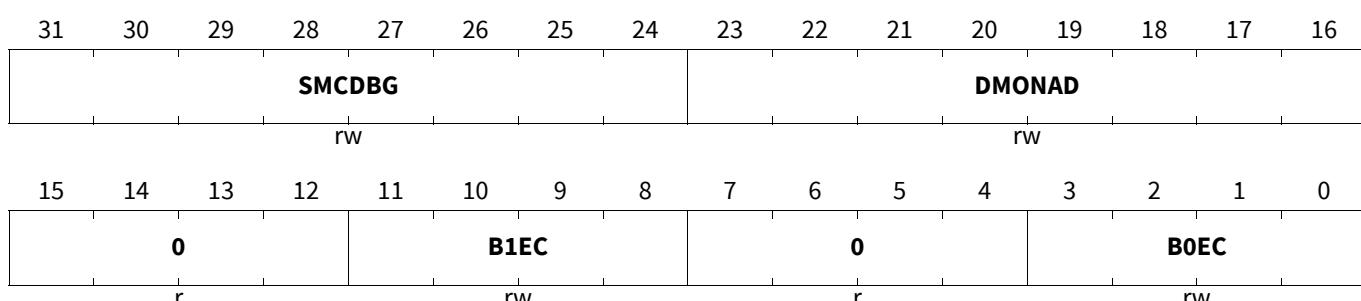
### OCDS Trigger Set Control 1 Register

#### OTSC1

#### OCDS Trigger Set Control 1 Register

(01E8<sub>H</sub>)

Reset Value: [Table 377](#)



## Power Management System (PMS)

Field	Bits	Type	Description
B0EC	3:0	rw	<b>OTGB0_TS16_EVRCON</b> 0 <sub>H</sub> No Module selected 1 <sub>H</sub> EVRCDPWM 2 <sub>H</sub> EVRCOUT 3 <sub>H</sub> EVR33OUT 4 <sub>H</sub> WUTCNT 5 <sub>H</sub> TCSCRINT <b>others</b> , reserved
B1EC	11:8	rw	<b>OTGB1_TS16_EVRCON</b> 0 <sub>H</sub> No Module selected 1 <sub>H</sub> EVRCDPWM 2 <sub>H</sub> EVRCOUT 3 <sub>H</sub> EVR33OUT 4 <sub>H</sub> WUTCNT 5 <sub>H</sub> TCSCRINT <b>others</b> , reserved
DMONAD	23:16	rw	<b>OTGB0_TS16_EVRCON DMONAD</b> The multiplexer signal selection documented in DMONAD coding table.
SMCDBG	31:24	rw	<b>OTGB0_TS16_EVRCON SMCDBG</b> Reserved for future extensions.
0	7:4, 15:12	r	<b>Reserved</b> Read as 0; must be written with 0.

**Table 377 Reset Values of OTSC1**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Debug Reset	0000 0000 <sub>H</sub>	

**Access Enable Register 0**

The Access Enable Register 0 restricts write access to all PMS registers so that they may only be written by specified bus masters (e.g. CPUs). See the Bus chapter for the mapping of TAG ID to specific system masters and CPUs).

**ACCENO**

(01FC <sub>H</sub> )																Application Reset Value: FFFF FFFF <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw					

## Power Management System (PMS)

Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the PMS kernel addresses for transactions with the Master TAG ID n 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

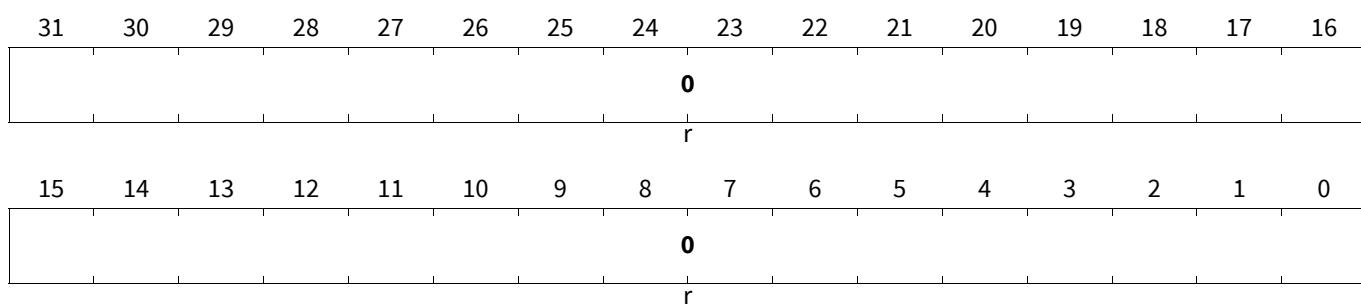
### Access Enable Register 1

#### ACCEN1

#### Access Enable Register 1

(01F8<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>0</b>	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

### 11.3.1.6 SMU\_STDBY Registers

The following registers are specified in the SMU chapter of this book:

- AG2i\_STDBY (i=0)
- MONBISTSTAT
- MONBISTCTRL
- CMD\_STDBY
- AG2iFSP\_STDBY (i=0)

## Power Management System (PMS)

### 11.3.2 Power Management Control Registers (SCU)

**Table 378 Register Overview - PMC (sorted by Name)**

Short Name	Long Name	Offset Address	Page Number
DTSCLIM	Core Die Temperature Sensor Limit Register	0108 <sub>H</sub>	<a href="#">190</a>
DTSCSTAT	Core Die Temperature Sensor Status Register	0104 <sub>H</sub>	<a href="#">190</a>
PMCSR0	Power Management Control and Status Register	00C8 <sub>H</sub>	<a href="#">182</a>
PMCSR1	Power Management Control and Status Register	00CC <sub>H</sub>	<a href="#">183</a>
PMCSR2	Power Management Control and Status Register	00D0 <sub>H</sub>	<a href="#">184</a>
PMCSR3	Power Management Control and Status Register	00D4 <sub>H</sub>	<a href="#">185</a>
PMCSR4	Power Management Control and Status Register	00D8 <sub>H</sub>	<a href="#">186</a>
PMCSR5	Power Management Control and Status Register	00DC <sub>H</sub>	<a href="#">187</a>
PMSTAT0	Power Management Status Register 0	00E4 <sub>H</sub>	<a href="#">180</a>
PMSWCR1	Standby and Wake-up Control Register 1	00E8 <sub>H</sub>	<a href="#">188</a>
PMTRCSR0	Power Management Transition Control and Status Register 0	0198 <sub>H</sub>	<a href="#">192</a>
PMTRCSR1	Power Management Transition Control and Status Register 1	019C <sub>H</sub>	<a href="#">194</a>
PMTRCSR2	Power Management Transition Control and Status Register 2	01A0 <sub>H</sub>	<a href="#">195</a>
PMTRCSR3	Power Management Transition Control and Status Register 3	01A4 <sub>H</sub>	<a href="#">196</a>

#### 11.3.2.1 Power Management Control and Status Registers

This section describes the kernel registers of the PMS module in SCU address space. Most of PMS kernel register names described in this section will be referenced in other parts of the Target Specification by the module name prefix “SCU\_”. The set of registers used for Power Management control the issue of power modes, manage wake-up configuration and provide status information on mode transitions and modules. The request for Idle, Sleep or Standby mode is issued via PMCSR<sub>x</sub> registers.

Tricore atomic instructions (LDMST, ST.T, SWAP.W, SWAPMASK.W, CMPSWAP.W) only write back bits that are changing their level. This leads to the fact that bits that are already set cannot be written with a 1 when using RMW instructions. No problem exists when using direct write instructions (e.g. ST.W). This affects the status bits in register DTSCSTAT.LLU, UOF and INT bits which are cleared by writing 1s.

##### Power Management Status Register 0

###### PMSTAT0

Power Management Status Register 0 ( <a href="#">00E4<sub>H</sub></a> ) Application Reset Value: 0000 0001 <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

0

r															

CPU3L  
S

rh															

CPU2L  
S

rh															

CPU1L  
S

rh															

CPU0L  
S

rh															

0

r															

CPU5  
CPU4  
CPU3  
CPU2  
CPU1  
CPU0

rh															

## Power Management System (PMS)

Field	Bits	Type	Description
CPU0	0	rh	<b>CPU0 Status</b> This bit field reflects the current status of CPU0. 0 <sub>B</sub> CPU0 is in Halt or Idle Mode 1 <sub>B</sub> CPU0 is in Normal Run Mode
CPU1	1	rh	<b>CPU1 Status</b> This bit field reflects the current status of CPU1. 0 <sub>B</sub> CPU1 is in Halt or Idle Mode 1 <sub>B</sub> CPU1 is in Normal Run Mode
CPU2	2	rh	<b>CPU2 Status</b> This bit field reflects the current status of CPU2. 0 <sub>B</sub> CPU2 is in Halt or Idle Mode 1 <sub>B</sub> CPU2 is in Normal Run Mode
CPU3	3	rh	<b>CPU3 Status</b> This bit field reflects the current status of CPU3. 0 <sub>B</sub> CPU3 is in Halt or Idle Mode 1 <sub>B</sub> CPU3 is in Normal Run Mode
CPU4	4	rh	<b>CPU4 Status</b> This bit field reflects the current status of CPU4. 0 <sub>B</sub> CPU4 is in Halt or Idle Mode 1 <sub>B</sub> CPU4 is in Normal Run Mode
CPU5	5	rh	<b>CPU5 Status</b> This bit field reflects the current status of CPU5. 0 <sub>B</sub> CPU5 is in Halt or Idle Mode 1 <sub>B</sub> CPU5 is in Normal Run Mode
CPU0LS	16	rh	<b>CPU0LS Status</b> This bit field reflects the current status of CPU0 Lockstep Checker Core. The activation of the Lockstep is configured in UCB BMI configuration and determines the default reset value. The default reset value 0 is for the case where CPU0LS is disabled in UCB BMI configuration. 0 <sub>B</sub> CPU0LS is disabled or in Halt or Idle Mode 1 <sub>B</sub> CPU0LS is enabled and in Normal Run Mode
CPU1LS	17	rh	<b>CPU1LS Status</b> This bit field reflects the current status of CPU1 Lockstep Checker Core. The activation of the Lockstep is configured in UCB BMI configuration and determines the default status. The default reset value 0 is for the case where CPU1LS is disabled in UCB BMI configuration. 0 <sub>B</sub> CPU1LS is disabled or in Halt or Idle Mode 1 <sub>B</sub> CPU1LS is enabled and in Normal Run Mode

## Power Management System (PMS)

Field	Bits	Type	Description
CPU2LS	18	rh	<b>CPU2LS Status</b> This bit field reflects the current status of CPU2 Lockstep Checker Core. The activation of the Lockstep is configured in UCB BMI configuration and determines the default status. The default reset value 0 is for the case where CPU0LS is disabled in UCB BMI configuration. 0 <sub>B</sub> CPU2LS is disabled or in Halt or Idle Mode 1 <sub>B</sub> CPU2LS is enabled and in Normal Run Mode
CPU3LS	19	rh	<b>CPU3LS Status</b> This bit field reflects the current status of CPU3 Lockstep Checker Core. The activation of the Lockstep is configured in UCB BMI configuration and determines the default status. The default reset value 0 is for the case where CPU0LS is disabled in UCB BMI configuration. 0 <sub>B</sub> CPU3LS is disabled or in Halt or Idle Mode 1 <sub>B</sub> CPU3LS is enabled and in Normal Run Mode
0	15:6, 31:20	r	<b>Reserved</b> Read as 0; should be written with 0.

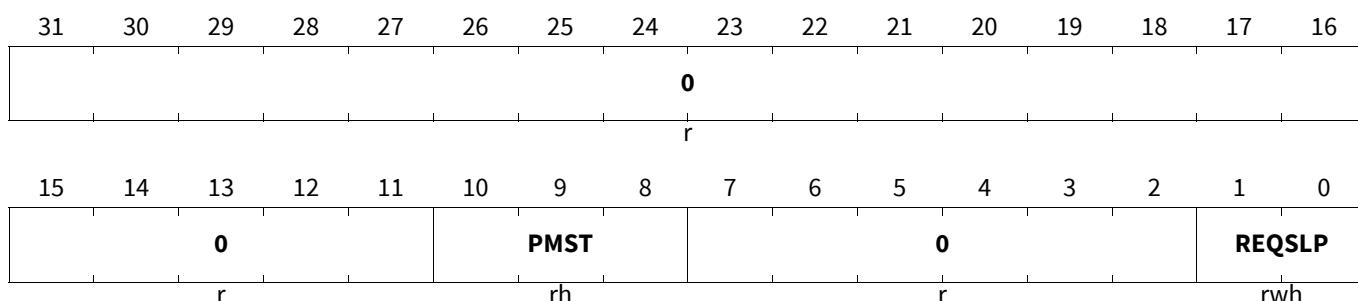
## Power Management Control and Status Register

Power Management Control and Status Register for CPU0

### PMCSR0

#### Power Management Control and Status Register(00C8<sub>H</sub>)

Application Reset Value: 0000 0100<sub>H</sub>



Field	Bits	Type	Description
REQSLP	1:0	rwh	<b>Idle Mode and Sleep Mode Request</b> In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUXSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. REQSLP maybe written only when either CPU or Safety ENDINIT bits are set to 0. CPU ENDINIT bit has to be set back after REQSLP is written for the mode transition to take place. In case of Safety ENDINIT, the mode transition will be issued immediately and does not wait till Safety ENDINIT is set back to 1 again. 00 <sub>B</sub> Request CPU Run Mode 01 <sub>B</sub> Request CPU Idle Mode 10 <sub>B</sub> Request System Sleep Mode 11 <sub>B</sub> Request System Standby Mode

## Power Management System (PMS)

Field	Bits	Type	Description
<b>PMST</b>	10:8	rh	<b>Power management Status</b> This bit field reflects the current status of the CPU. 000 <sub>B</sub> Reserved, do not use this combination 001 <sub>B</sub> Normal Run Mode <sup>1)</sup> 010 <sub>B</sub> CPU Idle Mode requested 011 <sub>B</sub> CPU Idle Mode acknowledged 100 <sub>B</sub> Sleep Mode requested 101 <sub>B</sub> Reserved, do not use this combination 110 <sub>B</sub> Standby Mode requested 111 <sub>B</sub> Reserved, do not use this combination
<b>0</b>	7:2, 31:11	r	<b>Reserved</b> Read as 0; should be written with 0.

- 1) After a reset, all CPUs are in “Normal Run Mode”, but this does not mean that all CPUs are executing code. This mode also includes the CPU “halt” mode which is the start-up default for all except CPU0.

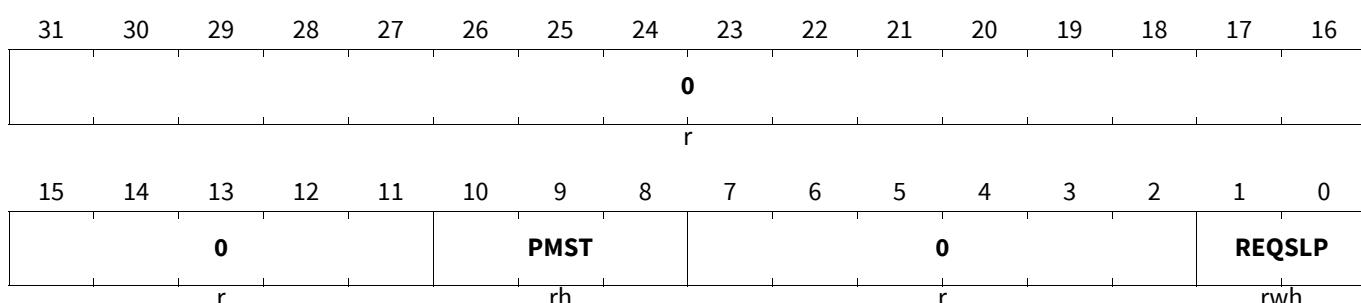
## Power Management Control and Status Register

Power Management Control and Status Register for CPU1. On product variants where CPU1 is not available, this register has no function.

### PMCSR1

#### Power Management Control and Status Register(00CC<sub>H</sub>)

Application Reset Value: 0000 0100<sub>H</sub>



Field	Bits	Type	Description
<b>REQSLP</b>	1:0	rwh	<b>Idle Mode and Sleep Mode Request</b> In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. REQSLP maybe written only when either CPU or Safety ENDINIT bits are set to 0. CPU ENDINIT bit has to be set back after REQSLP is written for the mode transition to take place. In case of Safety ENDINIT, the mode transition will be issued immediately and does not wait till Safety ENDINIT is set back to 1 again. 00 <sub>B</sub> Request CPU Run Mode 01 <sub>B</sub> Request CPU Idle Mode 10 <sub>B</sub> Request System Sleep Mode 11 <sub>B</sub> Request System Standby Mode

## Power Management System (PMS)

Field	Bits	Type	Description
<b>PMST</b>	10:8	rh	<b>Power management Status</b> This bit field reflects the current status of the CPU. 000 <sub>B</sub> Reserved, do not use this combination 001 <sub>B</sub> Normal Run Mode <sup>1)</sup> 010 <sub>B</sub> CPU Idle Mode requested 011 <sub>B</sub> CPU Idle Mode acknowledged 100 <sub>B</sub> Sleep Mode requested 101 <sub>B</sub> Reserved, do not use this combination 110 <sub>B</sub> Standby Mode requested 111 <sub>B</sub> Reserved, do not use this combination
<b>0</b>	7:2, 31:11	r	<b>Reserved</b> Read as 0; should be written with 0.

- 1) After a reset, all CPUs are in “Normal Run Mode”, but this does not mean that all CPUs are executing code. This mode also includes the CPU “halt” mode which is the start-up default for all except CPU0.

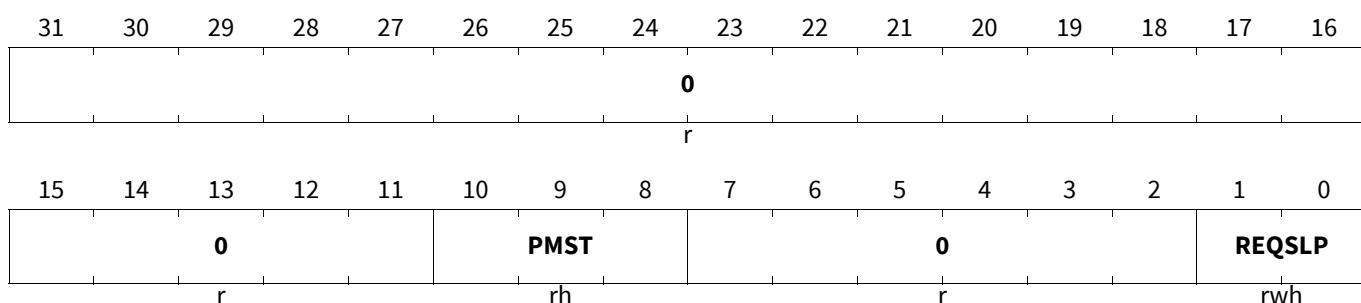
## Power Management Control and Status Register

Power Management Control and Status Register for CPU2. On product variants where CPU2 is not available, this register has no function.

### PMCSR2

#### Power Management Control and Status Register(00D0<sub>H</sub>)

Application Reset Value: 0000 0100<sub>H</sub>



Field	Bits	Type	Description
<b>REQSLP</b>	1:0	rwh	<b>Idle Mode and Sleep Mode Request</b> In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. REQSLP maybe written only when either CPU or Safety ENDINIT bits are set to 0. CPU ENDINIT bit has to be set back after REQSLP is written for the mode transition to take place. In case of Safety ENDINIT, the mode transition will be issued immediately and does not wait till Safety ENDINIT is set back to 1 again. 00 <sub>B</sub> Request CPU Run Mode 01 <sub>B</sub> Request CPU Idle Mode 10 <sub>B</sub> Request System Sleep Mode 11 <sub>B</sub> Request System Standby Mode

## Power Management System (PMS)

Field	Bits	Type	Description
<b>PMST</b>	10:8	rh	<b>Power management Status</b> This bit field reflects the current status of the CPU. 000 <sub>B</sub> Reserved, do not use this combination 001 <sub>B</sub> Normal Run Mode <sup>1)</sup> 010 <sub>B</sub> CPU Idle Mode requested 011 <sub>B</sub> CPU Idle Mode acknowledged 100 <sub>B</sub> Sleep Mode requested 101 <sub>B</sub> Reserved, do not use this combination 110 <sub>B</sub> Standby Mode requested 111 <sub>B</sub> Reserved, do not use this combination
<b>0</b>	7:2, 31:11	r	<b>Reserved</b> Read as 0; should be written with 0.

- 1) After a reset, all CPUs are in “Normal Run Mode”, but this does not mean that all CPUs are executing code. This mode also includes the CPU “halt” mode which is the start-up default for all except CPU0.

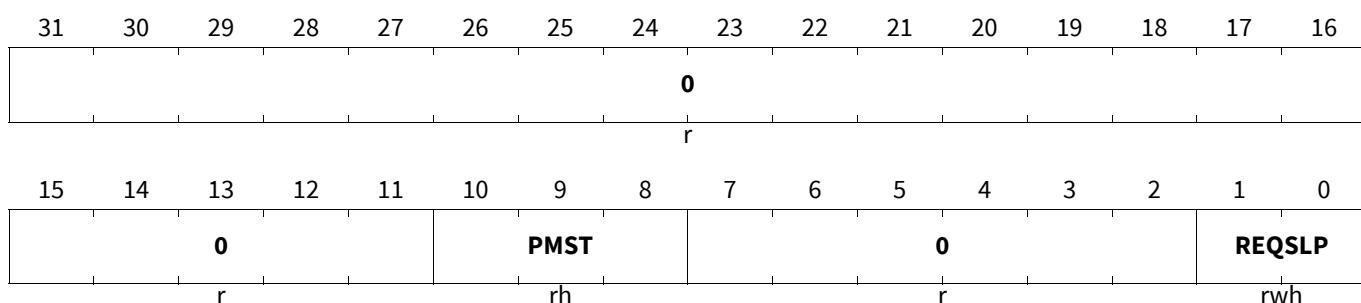
## Power Management Control and Status Register

Power Management Control and Status Register for CPU3. On product variants where CPU3 is not available, this register has no function.

### PMCSR3

#### Power Management Control and Status Register(00D4<sub>H</sub>)

Application Reset Value: 0000 0100<sub>H</sub>



Field	Bits	Type	Description
<b>REQSLP</b>	1:0	rwh	<b>Idle Mode and Sleep Mode Request</b> In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. REQSLP maybe written only when either CPU or Safety ENDINIT bits are set to 0. CPU ENDINIT bit has to be set back after REQSLP is written for the mode transition to take place. In case of Safety ENDINIT, the mode transition will be issued immediately and does not wait till Safety ENDINIT is set back to 1 again. 00 <sub>B</sub> Request CPU Run Mode 01 <sub>B</sub> Request CPU Idle Mode 10 <sub>B</sub> Request System Sleep Mode 11 <sub>B</sub> Request System Standby Mode

## Power Management System (PMS)

Field	Bits	Type	Description
<b>PMST</b>	10:8	rh	<b>Power management Status</b> This bit field reflects the current status of the CPU. 000 <sub>B</sub> Reserved, do not use this combination 001 <sub>B</sub> Normal Run Mode <sup>1)</sup> 010 <sub>B</sub> CPU Idle Mode requested 011 <sub>B</sub> CPU Idle Mode acknowledged 100 <sub>B</sub> Sleep Mode requested 101 <sub>B</sub> Reserved, do not use this combination 110 <sub>B</sub> Standby Mode requested 111 <sub>B</sub> Reserved, do not use this combination
<b>0</b>	7:2, 31:11	r	<b>Reserved</b> Read as 0; should be written with 0.

- 1) After a reset, all CPUs are in “Normal Run Mode”, but this does not mean that all CPUs are executing code. This mode also includes the CPU “halt” mode which is the start-up default for all except CPU0.

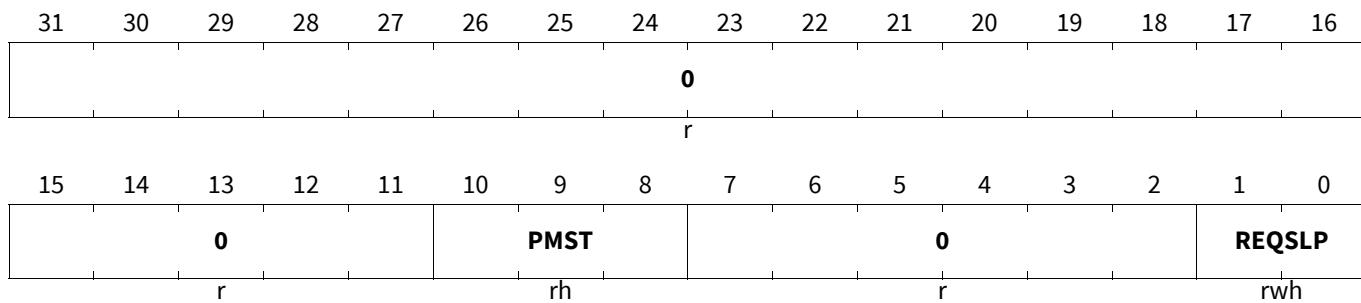
## Power Management Control and Status Register

Power Management Control and Status Register for CPU4. On product variants where CPU4 is not available, this register has no function.

### PMCSR4

#### Power Management Control and Status Register(00D8<sub>H</sub>)

Application Reset Value: 0000 0100<sub>H</sub>



Field	Bits	Type	Description
<b>REQSLP</b>	1:0	rwh	<b>Idle Mode and Sleep Mode Request</b> In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. REQSLP maybe written only when either CPU or Safety ENDINIT bits are set to 0. CPU ENDINIT bit has to be set back after REQSLP is written for the mode transition to take place. In case of Safety ENDINIT, the mode transition will be issued immediately and does not wait till Safety ENDINIT is set back to 1 again. 00 <sub>B</sub> Request CPU Run Mode 01 <sub>B</sub> Request CPU Idle Mode 10 <sub>B</sub> Request System Sleep Mode 11 <sub>B</sub> Request System Standby Mode

## Power Management System (PMS)

Field	Bits	Type	Description
<b>PMST</b>	10:8	rh	<b>Power management Status</b> This bit field reflects the current status of the CPU. 000 <sub>B</sub> Reserved, do not use this combination 001 <sub>B</sub> Normal Run Mode <sup>1)</sup> 010 <sub>B</sub> CPU Idle Mode requested 011 <sub>B</sub> CPU Idle Mode acknowledged 100 <sub>B</sub> Sleep Mode requested 101 <sub>B</sub> Reserved, do not use this combination 110 <sub>B</sub> Standby Mode requested 111 <sub>B</sub> Reserved, do not use this combination
<b>0</b>	7:2, 31:11	r	<b>Reserved</b> Read as 0; should be written with 0.

- 1) After a reset, all CPUs are in “Normal Run Mode”, but this does not mean that all CPUs are executing code. This mode also includes the CPU “halt” mode which is the start-up default for all except CPU0.

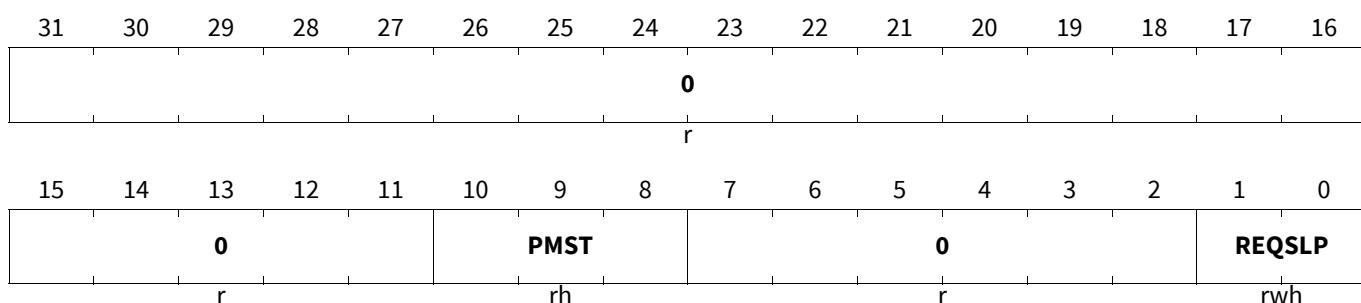
## Power Management Control and Status Register

Power Management Control and Status Register for CPU5. On product variants where CPU5 is not available, this register has no function.

### PMCSR5

#### Power Management Control and Status Register(00DC<sub>H</sub>)

Application Reset Value: 0000 0100<sub>H</sub>



Field	Bits	Type	Description
<b>REQSLP</b>	1:0	rwh	<b>Idle Mode and Sleep Mode Request</b> In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. REQSLP maybe written only when either CPU or Safety ENDINIT bits are set to 0. CPU ENDINIT bit has to be set back after REQSLP is written for the mode transition to take place. In case of Safety ENDINIT, the mode transition will be issued immediately and does not wait till Safety ENDINIT is set back to 1 again. 00 <sub>B</sub> Request CPU Run Mode 01 <sub>B</sub> Request CPU Idle Mode 10 <sub>B</sub> Request System Sleep Mode 11 <sub>B</sub> Request System Standby Mode

## Power Management System (PMS)

Field	Bits	Type	Description
PMST	10:8	rh	<b>Power management Status</b> This bit field reflects the current status of the CPU. 000 <sub>B</sub> Reserved, do not use this combination 001 <sub>B</sub> Normal Run Mode <sup>1)</sup> 010 <sub>B</sub> CPU Idle Mode requested 011 <sub>B</sub> CPU Idle Mode acknowledged 100 <sub>B</sub> Sleep Mode requested 101 <sub>B</sub> Reserved, do not use this combination 110 <sub>B</sub> Standby Mode requested 111 <sub>B</sub> Reserved, do not use this combination
0	7:2, 31:11	r	<b>Reserved</b> Read as 0; should be written with 0.

- 1) After a reset, all CPUs are in “Normal Run Mode”, but this does not mean that all CPUs are executing code. This mode also includes the CPU “halt” mode which is the start-up default for all except CPU0.

## Standby and Wake-up Control Register 1

### PMSWCR1

**Standby and Wake-up Control Register 1 (00E8<sub>H</sub>) Cold PowerOn Reset Value: 0100 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	<b>STBYEV</b>			<b>STBYEVEN</b>	<b>CPUSEL</b>			0							r
r	rw	w		rw											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	<b>IRADIS</b>		0	<b>CPUIDLSEL</b>			0							r	
r	rw	r		rw											

Field	Bits	Type	Description
CPUIDLSEL	10:8	rw	<b>CPU selection for Idle mode</b> This bit field allows a CPUx to issue Idle request to other CPUs in addition to itself. A request for Idle via PMCSR <sub>x</sub> .REQSLP=01 by CPUx will also trigger Idle requests to all other CPUs. <i>Note:</i> All other CPUIDLSEL bit combinations are reserved. 000 <sub>B</sub> Entry to the respective Idle mode is decided by each individual CPU. 001 <sub>B</sub> CPU0 Idle request will send all CPUs in Idle. 010 <sub>B</sub> CPU1 Idle request will send all CPUs in Idle. 011 <sub>B</sub> CPU2 Idle request will send all CPUs in Idle. 100 <sub>B</sub> CPU3 Idle request will send all CPUs in Idle. 101 <sub>B</sub> CPU4 Idle request will send all CPUs in Idle. 110 <sub>B</sub> CPU5 Idle request will send all CPUs in Idle.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>IRADIS</b>	12	rw	<p><b>Idle-Request-Acknowledge Sequence Disable</b></p> <p>This bit enables SCU Idle Request Acknowledge sequence to all modules on Standby entry. IRADIS bit has no effect incase of Standby entry triggered via PWRWKEN register bit.</p> <p>This bit shall be set before Standby entry to disable Idle request acknowledge sequence so that standby request is not blocked by a pending reset idle request acknowledge sequence.</p> <p>0<sub>B</sub> Idle-Request-Acknowledge Sequence issued on Standby entry. 1<sub>B</sub> Idle-Request-Acknowledge Sequence skipped on Standby entry.</p>
<b>CPUSEL</b>	26:24	rw	<p><b>CPU selection for Sleep and Standby mode</b></p> <p><i>Note:</i> All other CPUSEL bit combinations are reserved.</p> <p>001<sub>B</sub> Only CPU0 can trigger power down modes. 010<sub>B</sub> Only CPU1 can trigger power down modes. 011<sub>B</sub> Only CPU2 can trigger power down modes. 100<sub>B</sub> Only CPU3 can trigger power down modes. 101<sub>B</sub> Only CPU4 can trigger power down modes. 110<sub>B</sub> Only CPU5 can trigger power down modes. 111<sub>B</sub> Entry to power down modes is unanimously decided by all the CPUs.</p>
<b>STBYEVEN</b>	27	w	<p><b>Standby Entry Event configuration enable</b></p> <p>0<sub>B</sub> Bit STBYEV is not updated. 1<sub>B</sub> Bit STBYEV can be updated.</p>
<b>STBYEV</b>	30:28	rw	<p><b>Standby Entry Event Configuration</b></p> <p><i>Note:</i> All other bit combinations are reserved.</p> <p>000<sub>B</sub> Standby Entry triggered by setting PMCSR<sub>x</sub>.REQSLP register bit (Default). 100<sub>B</sub> Standby Entry triggered on ESR1 / NMI assertion.</p>
<b>0</b>	7:0, 11, 23:13, 31	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

## Power Management System (PMS)

### Core Die Temperature Sensor Status Register

#### DTSCSTAT

Core Die Temperature Sensor Status Register (0104<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
rh															

Field	Bits	Type	Description
RESULT	11:0	rh	<b>Result of the DTSC Measurement</b> This bit field shows the result of the DTSC measurement. The value given is directly related to the die temperature and can be evaluated using the following formula. $T (\text{°C}) = [\text{RESULT} / \text{Gnom}] - 273.15$ $T (\text{°K}) = [\text{RESULT}] / \text{G\_nom}$ $\text{RESULT} = \text{G\_nom} * \{T (\text{°C}) + 273.15\} = \text{G\_nom} * T (\text{°K})$ $\text{G\_nom} = 7.505$
0	31:12	r	<b>Reserved</b> Read as 0.

### Core Die Temperature Sensor Limit Register

#### DTSCLIM

Core Die Temperature Sensor Limit Register (0108<sub>H</sub>)

Application Reset Value: 0CD8 06D6<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UOF INT 0 INTEN															
rwh rwh r rw rw rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLU EN BGPO K 0															
rwh rw rh r rw															

Field	Bits	Type	Description
LOWER	11:0	rw	<b>DTSC Lower Limit</b> This bit field defines the lower limit of the DTSC temperature check. The DTSC measurement result is compared against this value and if the measurement result is less than or equal to the configured LOWER bitfield value; flag LLU is set.

## Power Management System (PMS)

Field	Bits	Type	Description
<b>BGPOK</b>	13	rh	<b>DTSC Bandgap OK</b> This bitfield indicates that the bandgap reference for the Core Die Temperature Sensor (DTSC) is available and ok. 0 <sub>B</sub> DTSC Bandgap is not ok. 1 <sub>B</sub> DTSC Bandgap is ok.
<b>EN</b>	14	rw	<b>DTSC Enable</b> This bitfield enables the Core Die Temperature Sensor (DTSC). The bitfield is reset on an application reset. 0 <sub>B</sub> DTSC is disabled 1 <sub>B</sub> DTSC is enabled
<b>LLU</b>	15	rwh	<b>DTSC Lower Limit Underflow</b> When this bit is set the related SMU DTSC alarm trigger is generated. This bit has to be written with zero in order to clear it. Writing a one has no effect. This bit is set when a DTSC measurement is finished and the result is below the lower limit (i.e. DTSCLIM.LOWER). 0 <sub>B</sub> No temperature underflow was detected 1 <sub>B</sub> A temperature underflow was detected
<b>UPPER</b>	27:16	rw	<b>DTSC Upper Limit</b> This bit field defines the upper limit of the DTSC temperature check. The DTSC measurement result is compared against this value and if the measurement result is greater than or equal to the configured UPPER bitfield value; flag UOF is set.
<b>INTEN</b>	28	rw	<b>DTSC Interrupt Enable</b> This bitfield enables the Core Die Temperature Sensor (DTSC) interrupt. The bitfield is reset on an application reset. 0 <sub>B</sub> DTSC Interrupt is disabled 1 <sub>B</sub> DTSC Interrupt is enabled
<b>INT</b>	30	rwh	<b>DTSC Interrupt status flag</b> This bit is set when SMU DTSC interrupt is generated when a DTSC measurement is finished. This bit is cleared by writing a zero. Writing a one has no effect. 0 <sub>B</sub> No DTSC interrupt is generated 1 <sub>B</sub> DTSC interrupt is generated
<b>UOF</b>	31	rwh	<b>DTSC Upper Limit Overflow</b> When this bit is set, the related SMU DTSC alarm trigger is generated. This bit has to be written with zero in order to clear it. Writing a one has no effect. This bit is set when a DTSC measurement is finished and the result is exceeding the upper limit (i.e. DTSCLIM.UPPER). 0 <sub>B</sub> No temperature overflow was detected 1 <sub>B</sub> A temperature overflow was detected
<b>0</b>	12, 29	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System (PMS)

### Power Management Transition Control and Status Register 0

#### PMTRCSR0

**Power Management Transition Control and Status Register 0(0198<sub>H</sub>)**

**Cold PowerOn Reset Value: 0000**

**0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
										<b>VDTCL R</b>	<b>VDTST P</b>	<b>VDTST RT</b>	<b>VDTO VIEN</b>	<b>VDTO VEN</b>	<b>VDTEN</b>
0	LPSLP EN				0					w	rw	rwh	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										<b>LJTCL R</b>	<b>LJTST P</b>	<b>LJTST RT</b>	<b>LJTOV IEN</b>	<b>LJTOV EN</b>	<b>LJTEN</b>
	SDSTEP					0				w	rw	rwh	rw	rw	rw

Field	Bits	Type	Description
<b>LJTEN</b>	0	rw	<b>Load Jump Timer Enable</b> This bit field enables the usage of load jump timer. 0 <sub>B</sub> Load Jump Timer inactive 1 <sub>B</sub> Load Jump Timer active
<b>LJTOVEN</b>	1	rw	<b>Load Jump Timer Overflow Enable</b> This bit field enables the update of LJTOV status bit on timer overflow or time out. 0 <sub>B</sub> LJTOV bit is not updated on a Load Jump Timer overflow. 1 <sub>B</sub> LJTOV bit is updated on a Load Jump Timer overflow.
<b>LJTOVIEN</b>	2	rw	<b>Load Jump Timer Overflow Interrupt Enable</b> This bit field enables the activation of interrupt on timer overflow or time out. 0 <sub>B</sub> LJTOV interrupt is inactive. 1 <sub>B</sub> LJTOV interrupt is activated on a Load Jump Timer overflow.
<b>LJTSTRT</b>	3	rwh	<b>Load Jump Timer Start</b> This bit field starts Load jump timer. This is intended for test purposes. The LJTSTRT remains set on a write and is cleared when LJTOV bit is set if LJTOVEN bit is enabled. 0 <sub>B</sub> Load Jump Timer status not changed. 1 <sub>B</sub> Load Jump Timer started.
<b>LJTSTP</b>	4	rw	<b>Load Jump Timer Stop</b> This bit field stops Load jump timer. This is intended for test purposes. The LJTSTP remains set on a write and is to be explicitly cleared by software. The LJTSTP stops the counter at the current value and timer re-starts from that value when LJTSTP is cleared and LJTSTRT is set. 0 <sub>B</sub> Load Jump Timer status not changed. 1 <sub>B</sub> Load Jump Timer stopped.

## Power Management System (PMS)

Field	Bits	Type	Description
LJTCLR	5	w	<b>Load Jump Timer Clear</b> This bit field clear Load jump timer count. This is intended for test purposes. This bit resets LJT and clears LJTRUN if LJTEN bit is set. 0 <sub>B</sub> Load Jump Count status not changed. 1 <sub>B</sub> Load Jump Timer Count cleared.
SDSTEP	15:12	rw	<b>Droop Voltage Step(vdroop_step_i)</b> This bit field defines the voltage offset for droop compensation on a load jump to the EVRC setpoint value. The request is made via <b>PMTRCSR3.VDROOPREQ</b> on an anticipated load jump with a voltage offset equal to the SDSTEP x 5 mV. The droop step is a positive offset if VDROOPREQ = 01b and is a negative offset if VDROOPREQ = 10b and no offset is applied if VDROOPREQ = 00b. Maximum Droop = 80 mV.
VDTEN	16	rw	<b>Voltage Droop Timer Enable</b> This bit field enables the usage of Voltage Droop timer. 0 <sub>B</sub> Voltage Droop Timer inactive 1 <sub>B</sub> Voltage Droop Timer active
VDTOVEN	17	rw	<b>Voltage Droop Timer Overflow Enable</b> This bit field enables the update of VDTOV status bit on timer overflow or time out. 0 <sub>B</sub> VDTOV bit is not updated on a Voltage Droop Timer overflow. 1 <sub>B</sub> VDTOV bit is updated on a Voltage Droop Timer overflow.
VDTOVIEN	18	rw	<b>Voltage Droop Timer Overflow Interrupt Enable</b> This bit field enables the activation of interrupt on timer overflow or time out. 0 <sub>B</sub> VDTOV interrupt is inactive. 1 <sub>B</sub> VDTOV interrupt is activated on a Voltage Droop Timer overflow.
VDTSTRT	19	rwh	<b>Voltage Droop Timer Start</b> This bit field starts Voltage Droop timer. This is intended for test purposes. The VDTSTRT remains set on a write and is cleared when VDTOV bit is set if VDTOVEN bit is enabled. 0 <sub>B</sub> Voltage Droop Timer status not changed. 1 <sub>B</sub> Voltage Droop Timer started.
VDTSTP	20	rw	<b>Voltage Droop Timer Stop</b> This bit field stops Voltage Droop timer. SCU cancels the droop request via signal sd_droop_cntr_i = 00. This is intended for test purposes. The VDTSTP remains set on a write and is to be explicitly cleared by software. The VDTSTP stops the counter at the current value and timer re-starts from that value when VDTSTP is cleared and VDTSTRT is set. 0 <sub>B</sub> Voltage Droop Timer status not changed. 1 <sub>B</sub> Voltage Droop Timer stopped.
VDTCLR	21	w	<b>Voltage Droop Timer Clear</b> This bit field clear Voltage Droop timer count. This is intended for test purposes. This bit resets VDT and clears VDTRUN if VDTEN bit is set. 0 <sub>B</sub> Voltage Droop Count status not changed. 1 <sub>B</sub> Voltage Droop Timer Count cleared.

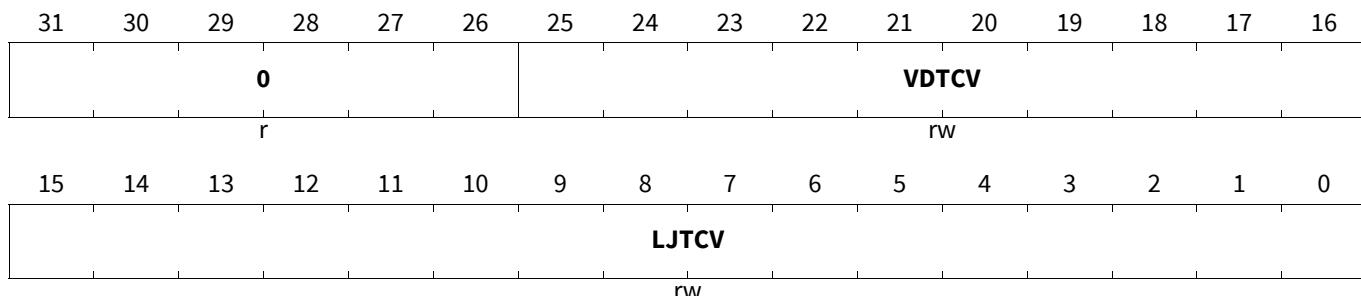
## Power Management System (PMS)

Field	Bits	Type	Description
LPSLPEN	29	rw	<b>EVRC Low Power Mode activation on a Sleep Request</b> PMS: This bit field enables the activation of LPM EVRC mode on a sleep request. PMSLE: Reserved, no function (no LPM for SC-DCDC EVRC). 0 <sub>B</sub> PMS: EVRC remains in normal operation mode during and after a sleep request. PMSLE: Reserved. 1 <sub>B</sub> PMS: LPM mode activated on a sleep request. PMSLE: Reserved.
0	11:6, 28:22, 31:30	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management Transition Control and Status Register 1

### PMTRCSR1

**Power Management Transition Control and Status Register 1(019C<sub>H</sub>)**      **Cold PowerOn Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
LJTCV	15:0	rw	<b>Load Jump Timer Compare Setpoint Value</b> This bit field defines the compare setpoint value of Load Jump timer. The compare event would lead to LJTOV bit being set and LJT interrupt being raised. The LJTRUN status bit, LDJMPREQ bit and LJTCNT value is reset to 0 on a compare event. X us is the compare value. LSB =1 us. Total range = 65.5 ms
VDTCV	25:16	rw	<b>Voltage Droop Timer Compare Setpoint Value</b> This bit field defines the compare setpoint value of Voltage Droop timer. The compare event would lead to VDTOV bit being set and VDT interrupt being raised. The VDTRUN status bit, VDROOPREQ bit and VDTCNT value is reset to 0 on a compare event. X us is the compare value. LSB =1 us. Total range = 1023 us
0	31:26	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System (PMS)

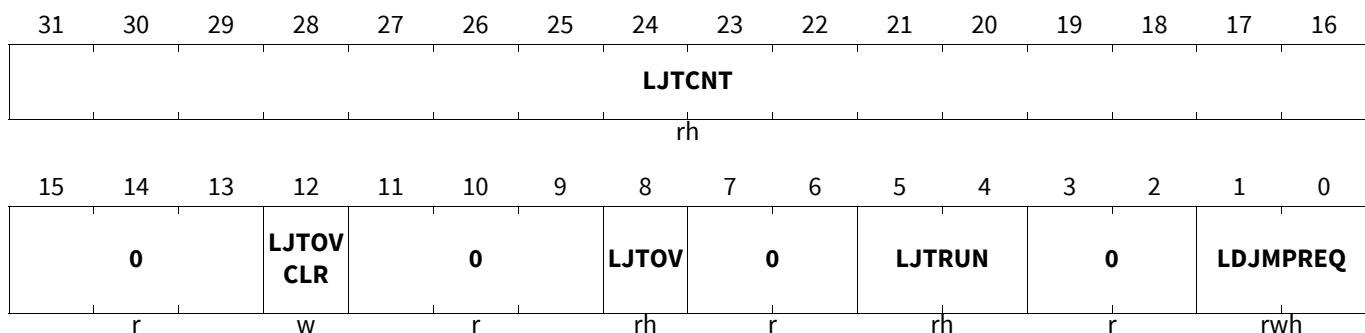
### Power Management Transition Control and Status Register 2

#### PMTRCSR2

**Power Management Transition Control and Status Register 2(01A0<sub>H</sub>)**

**Cold PowerOn Reset Value: 0000**

**0000<sub>H</sub>**



Field	Bits	Type	Description
<b>LDJMPREQ</b>	1:0	rwh	<p><b>Load Jump Request</b></p> <p>This bit requests a Load Jump consequently leading to Load Jump Timer start and LJTRUN bit being set if LJTEN=1. The request is not taken if LJTRUN bit is already in set state and LJT is currently running. The request is not taken if VDTRUN bit is already in set state and VDT is currently running. The request is also not taken if (LJTOV bit is set AND LJTOVEN bit is enabled). The request is also not taken if (VDTOV bit is set AND VDTOVEN bit is enabled). The LDJMPREQ bit is cleared on a compare overflow.</p> <p><i>Note:</i> All other bit combinations are reserved.</p> <p>00<sub>B</sub> Load Jump Timer inactive 01<sub>B</sub> Load Jump Request made and taken. Load Jump Timer activated.</p>
<b>LJTRUN</b>	5:4	rh	<p><b>Load Jump Timer Run Status</b></p> <p>This status bit indicates that the Load Jump timer is currently running and a Load Jump is currently taking place. The LJTRUN bit is cleared on a compare overflow.</p> <p><i>Note:</i> All other bit combinations are reserved.</p> <p>00<sub>B</sub> Load Jump and Load Jump Timer inactive 01<sub>B</sub> A SW triggered Load Jump active and Load Jump Timer active 10<sub>B</sub> A HW triggered Load Jump active and Load Jump Timer active (reserved for future)</p>

## Power Management System (PMS)

Field	Bits	Type	Description
LJTOV	8	rh	<b>Load Jump Timer Overflow Status</b> This status bit indicates that the Load Jump timer compare match has happened. If LJTOVEN bit is enabled, then LJTOV can only be cleared explicitly via LJTOVCLR bit. If LJTOVEN bit is disabled, LJTOV is cleared on a taken Load Jump Request (A new Load Jump request is taken only if both LJT & VDT are not currently running and no active Load Jump request is being processed). LJTOV being set will lead to an interrupt if LJTOVIEN is enabled. 0 <sub>B</sub> Load Jump Timer compare overflow has not happened. 1 <sub>B</sub> Load Jump Timer compare overflow has happened.
LJTOVCLR	12	w	<b>Load Jump Timer Overflow Status Clear</b> This bit clears LJTOV status bit and sets VDROOPREQ and LDJMPREQ to 0 if LJTOVEN bit is enabled. This bit always reads as 0. 0 <sub>B</sub> This clear bit has no effect on Load Jump Timer overflow flag. 1 <sub>B</sub> Load Jump Timer overflow flag is cleared.
LJTCNT	31:16	rh	<b>Load Jump Timer Value</b> This bit field reflects the current Load Jump timer value. LJTCNT value is cleared on timer overflow and on a taken Load Jump Request X us is the compare value. LSB = 1 us. Total range = 65.5 ms
0	3:2, 7:6, 11:9, 15:13	r	<b>Reserved</b> Read as 0; should be written with 0.

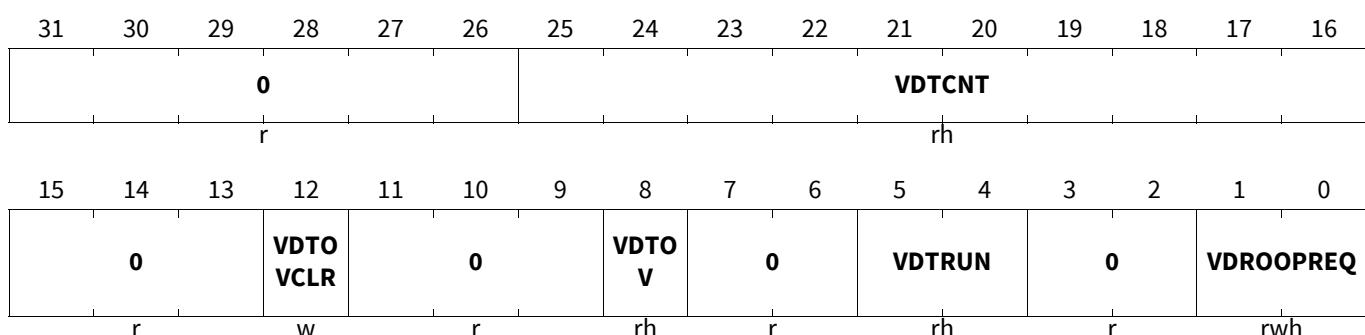
## Power Management Transition Control and Status Register 3

### PMTRCSR3

#### Power Management Transition Control and Status Register 3(01A4<sub>H</sub>)

Cold PowerOn Reset Value: 0000

0000<sub>H</sub>



## Power Management System (PMS)

Field	Bits	Type	Description
<b>VDROOPREQ</b>	1:0	rwh	<p><b>Voltage Droop Request</b></p> <p>This bit requests a Voltage Droop consequently leading to Voltage Droop Timer start and VDTRUN bit being set if VDTEN=1. The request is not taken if VDTRUN bit is already in set state and VDT is currently running. The request is also not taken if (VDTOV bit is set AND VDTOVEN bit is enabled). The droop step is a positive offset if sd_droop_cntr_i = 01 and is a negative offset if sd_droop_cntr_i = 10 and no offset is applied if sd_droop_cntr_i = 00 and is applied immediately.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Voltage Droop and Voltage Droop Timer inactive</li> <li>01<sub>B</sub> A Positive Voltage Droop Request made and taken. Voltage Droop Timer activated.</li> <li>10<sub>B</sub> A Negative Voltage Droop Request made and taken. Voltage Droop Timer activated.</li> <li>11<sub>B</sub> Voltage Droop and Voltage Droop Timer inactive</li> </ul>
<b>VDTRUN</b>	5:4	rh	<p><b>Voltage Droop Timer Run Status</b></p> <p>This status bit indicates that the Voltage Droop timer is currently running and a Voltage Droop is currently taking place. The VDTRUN bit is cleared on a compare overflow.</p> <p><i>Note:</i> All other bit combinations are reserved.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Voltage Droop and Voltage Droop Timer inactive</li> <li>01<sub>B</sub> A SW triggered Voltage Droop active and Voltage Droop Timer active</li> <li>10<sub>B</sub> A HW triggered Voltage Droop active and Voltage Droop Timer active (reserved for future)</li> </ul>
<b>VDTOV</b>	8	rh	<p><b>Voltage Droop Timer Overflow Status</b></p> <p>This status bit indicates that the Voltage Droop timer compare match has happened. If VDTOVEN bit is enabled, then VDTOV can only be cleared by explicitly via VDTOVCLR bit. If VDTOVEN bit is disabled, VDTOV is cleared on a taken Voltage Droop Request (A new Voltage Droop request is taken only if both LJT &amp; VDT are not currently running and no active Voltage Droop request is being processed). VDTOV being set will lead to an interrupt if VDTOVIEN is enabled. Incase SDVOK is set by EVRC before VDT compare match, VDTOV bit is not set.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> Voltage Droop Timer compare overflow has not happened.</li> <li>1<sub>B</sub> Voltage Droop Timer compare overflow has happened.</li> </ul>
<b>VDTOVCLR</b>	12	w	<p><b>Voltage Droop Timer Overflow Status Clear</b></p> <p>This bit clears VDTOV status bit if VDTOVEN bit is enabled. If VDTOVEN bit is disabled, this bit has no effect. This bit always reads as 0.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> This clear bit has no effect on Voltage Droop Timer overflow flag.</li> <li>1<sub>B</sub> Voltage Droop Timer overflow flag is cleared.</li> </ul>
<b>VDTCNT</b>	25:16	rh	<p><b>Voltage Droop Timer Value</b></p> <p>This bit field reflects the current Voltage Droop timer value. VDTCNT value is cleared on timer overflow and on a taken Voltage Droop Request. X us is the compare value. LSB = 1 us. Total range = 65.5 ms</p>

## Power Management System (PMS)

Field	Bits	Type	Description
0	3:2, 7:6, 11:9, 15:13, 31:26	r	<b>Reserved</b> Read as 0; should be written with 0.

### 11.4 IO Interfaces

The following table defines the signals connecting the PMS to other modules and the outside world.

Note that not all signals may be used in all members of the family. Consult the product specific appendix to see the available connections.

**Table 379 List of PMS Interface Signals**

Interface Signals	I/O	Description
HWCFG1IN	in	<b>HWCFG1 pin input</b> Hardware configuration 1 input for activation of EVR33 regulator.
HWCFG2IN	in	<b>HWCFG2 pin input</b> Hardware configuration 2 input for activation of EVRC regulator.
HWCFG4IN	in	<b>HWCFG4 pin input</b> Hardware configuration 4 input for test purposes.
HWCFG5IN	in	<b>HWCFG5 pin input</b> Hardware configuration 5 input for test purposes.
HWCFG6IN	in	<b>HWCFG6 pin input</b> Hardware configuration 6 input for activation of tri-state.
TESTMODEIN	in	<b>TESTMODE pin input</b> Test mode pin input to enable entry into test modes
PORSTIN	in	<b>PORST pin input</b> PORST pin input to trigger warm PORST
PORSTOUT	in	<b>PORST pin output</b> Cold PORST strong pull down control output to drive PORST pin low in case of primary monitor undervoltage events
PORSTWKPD	in	<b>PORST pad weak pull down control output</b> PORST weak pull down control to keep PORST weakly pulled low in case of bond wire breakage. The pull down is inactive to avoid additional current during STANDBY mode
ESR0PORST	out	<b>ESR0 control output during PORST activation</b> Warm PORST signal connected to ESR0 pad to ensure that ESR0 pad is asserted when PORST pin is asserted to propagate the reset
ESR0WKP	in	<b>ESR0 pin input</b> ESR0 pin input for wakeup from STANDBY mode
ESR1WKP	in	<b>ESR1 pin input</b> ESR1 pin input for wakeup from STANDBY mode

## Power Management System (PMS)

**Table 379 List of PMS Interface Signals (cont'd)**

Interface Signals	I/O	Description
PINAWKP	in	<b>PINA ( P14.1) pin input</b> P14.1 pin input for wakeup from STANDBY mode
PINBWKP	in	<b>PINB (P33.12) pin input</b> P33.12 pin input for wakeup from STANDBY mode
VGATE1P	out	<b>DCDC P ch. MOSFET gate driver output</b>
VGATE1N	out	<b>DCDC N ch. MOSFET gate driver output</b>
DCDCSYNCO	out	<b>DC-DC synchronization output</b>
WUTUFLOW	out	<b>WUT counter underflow signal to CCU6/GTM</b> WUT Underflow output to support WUT calibration
DCDCSYNCGTM	in	<b>DCDC synchronization signal input from GTM</b> Synchronisation input from GTM module to EVRC SMPS regulator
DCDCSYNCCU	in	<b>DCDC synchronization signal input from CCU6</b> Synchronisation input from CCU6 (CCU60 COUT63) module to EVRC SMPS regulator
VDDMLVL	out	<b>VDDM monitor signal to Converter</b> Signal indicating whether VDDM is supplied with 5V or 3.3V. 0: VDDM = 5V 1: VDDM = 3.3V.

## Power Management System (PMS)

### 11.5 Revision History

#### 11.5.1 Changes from AURIX 2G PMS V2.2.19 onwards

**Table 380 Revision History**

Reference	Change to Previous Version	Comment
V2.2.28		
<a href="#">Page 120</a>	Corrected VIN formula in the bit-field description of EVRSDSTAT0.ADCFBCV. ADCFBCV value includes trimming, thus subtracting to obtain VIN.	
<a href="#">Page 88</a>	Corrected VIN formula and related explanation in the bit-field descriptions of EVRADCSTAT.ADCCV, EVRADCSTAT.ADC33V, and EVRADCSTAT.ADCSWDV by removing the EVRTRIM2 ADC offset part (ADCOFFS does not need to be considered for the calculation of the output voltage, production test trimming makes sure to set the ADCOFFS correctly).	
<a href="#">Page 38</a>	Added a statement that the user shall not modify the default values of the EVRRSTCON register.	
<a href="#">Page 9</a>	Added to the description of the LVD reset release at T1 that VEVRSB is above the VLDRSTSB level.	
<a href="#">Page 11</a>	Added to the description of the LVD reset release at T1 that VEVRSB is above the VLDRSTSB level.	
<a href="#">Page 13</a>	Added to the description of the LVD reset release at T1 that VEVRSB is above the VLDRSTSB level.	
<a href="#">Page 15</a>	Added to the description of the LVD reset release at T1 that VEVRSB is above the VLDRSTSB level.	
<a href="#">Page 82</a>	Corrected list of registers with Safety Flip-Flops by removing EVRRSTHYS and DTSLIM registers.	
<a href="#">Page 38</a>	Enabled description of the HSMUVMON and HSMOVMON thresholds and documented that alarms are routed not only to the HSM module, but also to the SMU.	
<a href="#">Page 20</a>	Changed name of the DTS interrupt service request from SRC_DTS to SRC_PMSDTS, to match the name in the Interrupt Router specification.	
<a href="#">Page 78</a>	Enabled section on Core Die Temperature Sensor (DTSC).	
<a href="#">Page 47</a>	Corrected figure by adding DTSC interrupt event to SCR_SCUERU3 source and SRC_PMSDTS interrupt source with DTS interrupt event.	
<a href="#">Page 46</a>	Corrected name of the MONBIST control register for MONBIST enabling, by changing “MONBISTSTAT.TESTEN” to “MONBISTCTRL.TESTEN”	
<a href="#">Page 18</a>	Added statement that EVROSCCTRL register shall not be modified by application SW and that additional compensation can be enabled via EVROSCCTRL.OSCTEMPOFFS and EVROSCCTRL.OSCTRIMEN bits.	
<a href="#">Page 40</a>	Added statement that EVRUVMON2.VDDMLVLSEL bit-field shall not be modified by application SW and that its default value shall be kept with any updates of the undervoltage monitoring thresholds.	

## Power Management System (PMS)

**Table 380 Revision History (cont'd)**

Reference	Change to Previous Version	Comment
Page 18	Removed statement on measuring the SHPBG voltage on EVADC channel 29, as this is an internal feature not relevant for customer usage.	
<b>V2.2.29</b>		
Page 90	Updated note on Reset Values of EVRRSTCON register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 94	Updated note on Reset Values of EVRTRIM register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 100	Updated note on Reset Values of EVRMONCTRL register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 105	Updated note on Reset Values of EVRMONFILT register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 110	Updated note on Reset Values of EVRUVMON register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 112	Updated note on Reset Values of EVRUVMON2 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 111	Updated note on Reset Values of EVROVMON register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 114	Updated note on Reset Values of EVROVMON2 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 115	Updated note on Reset Values of HSMUVMON register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 117	Updated note on Reset Values of HSMOVMON register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 119	Updated note on Reset Values of EVROSCCTRL register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 121	Updated note on Reset Values of EVRSDCTRL0 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 123	Updated note on Reset Values of EVRSDCTRL1 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 128	Updated note on Reset Values of EVRSDCTRL2 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 130	Updated note on Reset Values of EVRSDCTRL3 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 134	Updated note on Reset Values of EVRSDCTRL4 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 135	Updated note on Reset Values of EVRSDCTRL5 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 136	Updated note on Reset Values of EVRSDCTRL6 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
Page 140	Updated note on Reset Values of EVRSDCTRL7 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	

## Power Management System (PMS)

**Table 380 Revision History (cont'd)**

Reference	Change to Previous Version	Comment
<a href="#">Page 146</a>	Updated note on Reset Values of EVRSDCTRL8 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 147</a>	Updated note on Reset Values of EVRSDCTRL9 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 148</a>	Updated note on Reset Values of EVRSDCTRL10 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 149</a>	Updated note on Reset Values of EVRSDCTRL11 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 124</a>	Updated note on Reset Values of EVRSDCOEFF0 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 126</a>	Updated note on Reset Values of EVRSDCOEFF1 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 131</a>	Updated note on Reset Values of EVRSDCOEFF2 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 132</a>	Updated note on Reset Values of EVRSDCOEFF3 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 137</a>	Updated note on Reset Values of EVRSDCOEFF4 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 139</a>	Updated note on Reset Values of EVRSDCOEFF5 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 142</a>	Updated note on Reset Values of EVRSDCOEFF6 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 143</a>	Updated note on Reset Values of EVRSDCOEFF7 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 144</a>	Updated note on Reset Values of EVRSDCOEFF8 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 145</a>	Updated note on Reset Values of EVRSDCOEFF9 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 3</a>	Added a description of the pull-up and pull-down device resistance range for HWCFG[6] and HWCFG[1,2] in order to recognize the high respectively low state setting.	
<a href="#">Page 192</a>	Updated the PMTRCSR0.LPSLPEN bit-field description, in the case of PMSLE it is reserved, since the SC-DCDC has no Low-Power Mode (LPM).	
<a href="#">Page 67</a>	Corrected the name of the SMUEN register, from CTRL.SMUEN into CMD_STDBY.SMUEN.	
<a href="#">Page 70</a>	Corrected the name of the SMUEN register, from CTRL.SMUEN into CMD_STDBY.SMUEN.	
<b>V2.2.30</b>		
<a href="#">Page 34</a>	Removed reference to capacitor component name, since it was discontinued. Only the capacitor value is specified, component choice according to data sheet parameters.	
<a href="#">Page 4</a>	Added VFLEX2 supply rail to <a href="#">Table 285</a> , present only on TC37xEXT silicon.	

## Power Management System (PMS)

**Table 380 Revision History (cont'd)**

Reference	Change to Previous Version	Comment
<a href="#">Page 5</a>	Added VFLEX2 supply rail to <a href="#">Table 286</a> , present only on TC37xEXT silicon.	
<a href="#">Page 6</a>	Added VFLEX2 supply rail to <a href="#">Table 287</a> , present only on TC37xEXT silicon.	
<a href="#">Page 6</a>	Added VFLEX2 supply rail to <a href="#">Table 288</a> , present only on TC37xEXT silicon.	
<a href="#">Page 7</a>	Added VFLEX2 supply rail to <a href="#">Figure 94</a> , present only on TC37xEXT silicon.	
<a href="#">Page 23</a>	Added VFLEX2 supply rail to <a href="#">Figure 103</a> , present only on TC37xEXT silicon.	
<a href="#">Page 25</a>	Added VFLEX2 supply rail to <a href="#">Figure 105</a> , present only on TC37xEXT silicon.	
<a href="#">Page 35</a>	Added VFLEX2 supply rail to <a href="#">Figure 107</a> , present only on TC37xEXT silicon.	
<a href="#">Page 39</a>	Enabled description for activating EVR33 short detection scheme.	
<a href="#">Page 39</a>	Enabled description of register for EVR33 short detection configuration.	
<a href="#">Page 59</a>	Removed question marks and updated PMSTAT0.CPU[y] & LS status bit values for the “System during Sleep Mode” condition.	
<a href="#">Page 182</a>	Updated description of <b>PMCSR0</b> register (for CPU0).	
<a href="#">Page 183</a>	Updated description of <b>PMCSR1</b> register, indicating that this register has no function if CPU1 is not available on a product variant.	
<a href="#">Page 184</a>	Updated description of <b>PMCSR2</b> register, indicating that this register has no function if CPU2 is not available on a product variant.	
<a href="#">Page 185</a>	Updated description of <b>PMCSR3</b> register, indicating that this register has no function if CPU3 is not available on a product variant.	
<a href="#">Page 186</a>	Updated description of <b>PMCSR4</b> register, indicating that this register has no function if CPU4 is not available on a product variant.	
<a href="#">Page 187</a>	Updated description of <b>PMCSR5</b> register, indicating that this register has no function if CPU5 is not available on a product variant.	
<a href="#">Page 34</a>	Added register update sequence for the 0.8MHz, IDD<700mA configuration, to <a href="#">Table 290</a> .	

### V2.2.31

<a href="#">Page 71</a>	Corrected the blanking filter minimum time: reduced from 1ms to 40us. There is no issue if a longer filter time has been configured (e.g. 1ms), but the minimum required duration is of only 40us.	
<a href="#">Page 45</a>	Corrected in <a href="#">Figure 113</a> the VDDPD secondary monitoring over-voltage and under-voltage levels.	
<a href="#">Page 31</a>	Updated EVRSDCOEFF0 setting to 0x360974B6 for improved stability of the regulator loop in Low-End configurations (IDD<500 mA)and fDCDC=1.8 MHz.	
<a href="#">Page 27</a>	Added statement that for Low-End configurations (IDD<500 mA), synchronization lock and unlock procedures shall not touch the bitfield EVRSDCOEFF0.M0SRMPCOEFF at all.	
<a href="#">Page 28</a>	Added statement that for Low-End configurations (IDD<500 mA), synchronization lock and unlock procedures shall not touch the bitfield EVRSDCOEFF0.M0SRMPCOEFF at all.	
<a href="#">Page 3</a>	Removed reference to EVR33 disabling bit.	
<a href="#">Page 35</a>	Removed reference to EVR33 disabling bit.	

## Power Management System (PMS)

**Table 380 Revision History (cont'd)**

Reference	Change to Previous Version	Comment
<a href="#">Page 3</a>	Added information about VEXT-buffered PU1 pads state after standby mode entry (regardless of the HWCFG[6] setting).	
<a href="#">Page 67</a>	Added information about VEXT-buffered PU1 pads state after standby mode entry (regardless of the PMSWCR5.TRISTEQ setting).	
<a href="#">Page 70</a>	Added information about VEXT-buffered PU1 pads state after standby mode entry (regardless of the PMSWCR5.TRISTEQ setting).	
<a href="#">Page 71</a>	Added information about VEXT-buffered PU1 pads state after standby mode entry (regardless of the PMSWCR5.TRISTEQ setting).	
<a href="#">Page 7</a>	Corrected VDDPD voltage in <a href="#">Figure 94</a> .	
<a href="#">Page 5</a>	Corrected VDDPD voltage range in <a href="#">Table 286</a> .	
<a href="#">Page 6</a>	Corrected VDDPD voltage range in <a href="#">Table 287</a> .	

### V2.2.32

<a href="#">Page 27</a>	Corrected the link to the register field, from <a href="#">EVRSDCTRL11</a> to <a href="#">EVRSDCTRL0</a> .	
<a href="#">Page 51</a>	Added statement to the Standby Mode (Only VEVRSB supplied).	
<a href="#">Page 60</a>	Added statement about Standby Entry trigger.	
<a href="#">Page 61</a>	Added statement about Standby Entry trigger.	
<a href="#">Page 67</a>	Added statement on Standby Entry trigger event and a standby entry trigger to the steps to enter standby.	
<a href="#">Page 67</a>	Added bullet list item.	
<a href="#">Page 70</a>	Added bullet list item.	
<a href="#">Page 51</a>	Changed bullet list item.	
<a href="#">Page 63</a>	Replaced sentence in <a href="#">Chapter 11.2.3.4.6</a> .	
<a href="#">Page 161</a>	For clarification reset value for "After SSW execution" added, no functional change.	

---

## Power Management System for Low-End (PMSLE)

### 12 Power Management System for Low-End (PMSLE)

This chapter describes Power Supply Generation and Power Management in TC3xx in following sections:

- Power Supply Infrastructure and Supply Start-up (see [Section 12.2.1](#))
  - Supply Mode Selection (see [Section 12.2.1.1](#))
  - Supply Ramp-up and Ramp-down Behavior (see [Section 12.2.1.2](#))
  - Independent Supply domain for Regulators and Monitors (see [Section 12.2.1.3.1](#))
  - Reference Voltage Generation (see [Section 12.2.1.3.2](#))
  - 100 MHz Back-up Clock (see [Section 12.2.1.3.3](#))
  - Die Temperature Measurement (see [Section 12.2.1.4](#))
- Power Supply Generation and Monitoring (see [Section 12.2.2](#))
  - VDDP3 Supply Generation
    - Linear Regulator Mode (EVR33) (see [Section 12.2.2.1](#))
    - External Supply Modes (see [Section 12.2.2.4](#))
  - VDD Supply Generation
    - Step-down Regulator (EVRC) (see [Section 12.2.2.2](#))
    - External Supply Modes (see [Section 12.2.2.4](#))
  - Supply Voltage Monitoring (see [Section 12.2.2.5](#))
    - Primary under-voltage monitors and Cold PORST (see [Section 12.2.2.5.1](#))
    - Secondary over- and under-voltage monitors and alarm generation (see [Section 12.2.2.5.2](#))
    - Built In Self Tests (PBIST and MONBIST) (see [Section 12.2.2.5.3](#) and [Section 12.2.2.5.4](#))
  - Interrupts (see [Section 12.2.2.6](#))
  - OCDS Interface (see [Section 12.2.2.7](#))
- Power Management (see [Section 12.2.3](#))
  - Idle Mode (see [Section 12.2.3.2](#))
  - Sleep Mode (see [Section 12.2.3.3](#))
  - Standby Mode (see [Section 12.2.3.4](#))
  - Wake-up Timer (WUT) (see [Section 12.2.3.4.7](#))
  - Standby ControlleR (SCR) Interface (see [Section 12.2.3.4.6](#))
  - Load Jump Sequencing and Voltage Droop (see [Section 12.2.3.5](#))
- Power Management System Register Tables
  - PMS Power Management Register Table (see [Page 80](#))
  - SCU Power Management Register Table (see [Page 175](#))

## Power Management System for Low-End (PMSLE)

### 12.1 Overview

On-chip linear and switch mode voltage regulators are implemented in TC3xx thereby enabling a single source power supply concept. The external nominal system supply from external regulator may be either 5 V or 3.3 V. The Embedded Voltage Regulators (EVR33 & EVRC) in turn generate the VDDP3 and VDD supply voltages required internally for the core, flash and port domains. EVRC regulator is implemented as a SMPS regulator and generates core supply either from 5 V or 3.3 V external supply. EVR33 regulator is implemented always as a LDO regulator and is required only in case of 5 V external supply.

Depending on the chosen EVR mode, the actual power consumption, EMI requirements and thermal constraints of the system; additional external components like MOSFETs, inductors and capacitors may be required. It is also possible to supply all voltages (VEXT, VDDP3 and VDD) externally ensuring compliance to the legacy supply concept.

All supply and generated voltages are monitored for brownout conditions by primary monitors setting the device into cold power-on reset state in case of violation. All supply and generated voltages are monitored again redundantly by secondary monitors against programmable over-voltage and under-voltage levels. If these levels are violated, either an interrupt or an alarm to the SMU may be generated.

All internal supplies except analog supplies (VAREFx & VDDM) may be supplied by the EVR33 & EVRC. The analog supply domain is separated from the main EVR supply domain and can be supplied by separate external regulators or trackers. It is possible to have a mixed supply scheme with a 5 V ADC domain (VAREFx = VDDM = 5 V) and the remaining system running on 3.3 V supply (VEXT = VDDP3 = 3.3 V).

### 12.2 Functional Description

#### 12.2.1 Power Supply Infrastructure and Supply Start-up

##### 12.2.1.1 Supply Mode Selection

The choice of the supply scheme at startup is based on the latched status of HWCFG[2:1] pins before cold PORST release and is indicated by [PMSWSTAT](#).HWCFGEVR status flags. Following supply modes are supported and are further enumerated in [Table 381](#).

- Single source 5 V supply level ( $VEXT = 5\text{ V}$ ) is supported in following topologies.
  - EVRC in SMPS mode with external flying capacitor and EVR33 in LDO mode with internal pass devices.
- Single source 3.3 V supply level ( $VEXT = VDDP3 = 3.3\text{ V}$ ) is supported in following topologies.
  - EVRC in SMPS mode with external flying capacitor and EVR33 is inactive.
- Supplies are provided externally and the respective EVRs are in disabled state.
  - 5 V ( $VEXT$ ) and 1.25 V ( $VDD$ ) supplied externally. EVR33 in LDO mode with internal pass devices.
  - 5 V ( $VEXT$ ) and 3.3 V ( $VDDP3$ ) supplied externally. EVRC in SMPS mode with external flying capacitor.
  - 5 V ( $VEXT$ ), 3.3 V ( $VDDP3$ ) and 1.25 V ( $VDD$ ) are all supplied externally.

EVRC is enabled or disabled at startup via the HWCFG[2] configuration pin. In case EVRC is selected, VCAP0 and VCAP1 pins shall be connected to an external flying capacitor as shown in [Figure 132](#).

## Power Management System for Low-End (PMSLE)

EVR33 is enabled or disabled at startup via the HWCFG[1] configuration pin. In case of single source 3.3 V supply, EVR33 is disabled and VDDx3 & VEXT pins are supplied externally by 3.3 V. EVR33 LDO uses internal pass devices distributed on the chip.

The allowed ranges of supplies among different supply rails during different power modes are documented in [Table 382](#) and [Table 383](#). The allowed combinations of nominal external supply voltages among different supply rails are documented in [Table 384](#). All externally provided supplies must be available and be stable before warm PORST reset release by external regulator(s).

HWCFG [2:1] are latched during supply ramp-up and the respective regulators are consequently started. The latched values are stored in PMSWSTAT.HWCFGEV register bits. The latched values are retained through a cold PORST and are only reset if EVR LVD (Low Voltage Detector) reset is asserted. HWCFG signals are filtered through a spike / glitch filter and are monitored for a constant level over a 28us - 115us nominal debouncing period before the value is considered as valid so as to ensure reliable operation in noisy environment. The current state of EVRs are reflected in EVRSTAT.EVRx3 flags. For small package variants, some of the HWCFG configuration pins may be absent and both EVRs are activated by default at startup.

HWCFG[6] pin is latched during early VEXT supply ramp-up ( $VEXT < VDDPPA$ ) to decide and set the default reset state of port pins as early as possible. During the initial ramp-up phase of VEXT and VEVRSB supply voltage from 0V up to VDDPPA limit, the voltage levels of pins are undefined till the transistor threshold voltages are reached. After VDDPPA limit, the pins behave as inputs with pull-up if HWCFG[6] = 1 or are in tristate if HWCFG[6] = 0. During the later stage of ramp-up, the latched HWCFG[6] value is stored in PMSWSTAT.TRIST register bit.

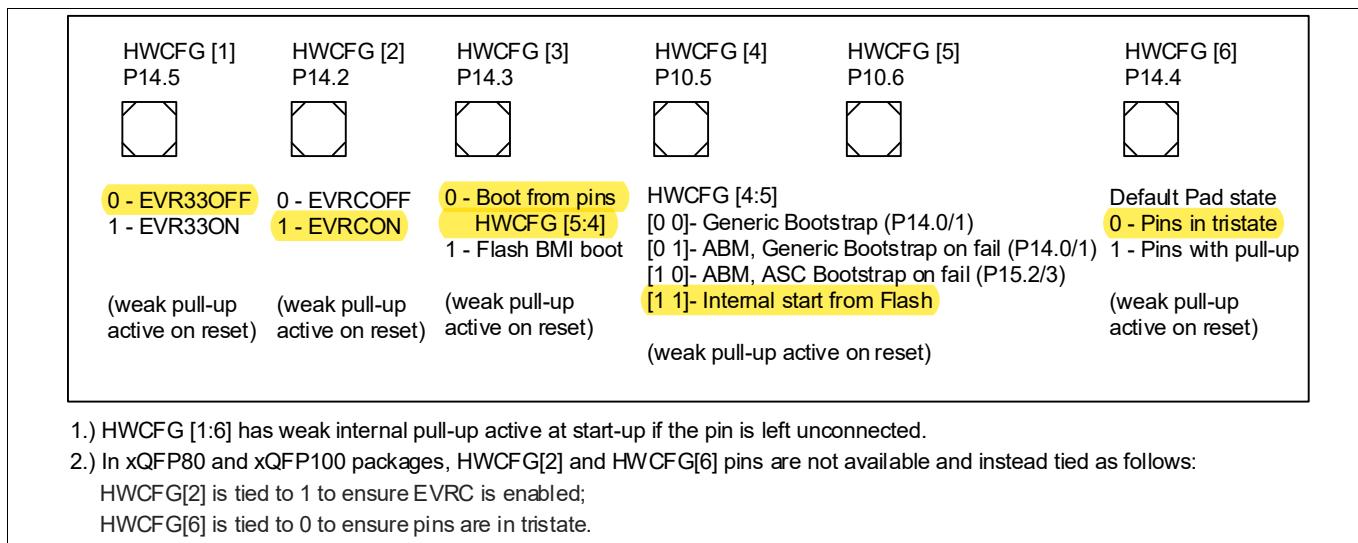
HWCFG [1,2,3,6] pins have weak internal pull-up active at start-up irrespective of HWCFG [6] pin level to ensure that the device boots with a defined configuration if HWCFG [1,2,3,6] pins are left unconnected. HWCFG [1,2,3,6] pins are only latched by the PMS on every initial supply ramp-up and are not re-latched during warm reset events (warm PORST, system or application resets) or on exit from Standby mode. All HWCFG pins are latched on internal reset release additionally (between 100us – 180us after warm reset assertion) and the status is stored redundantly in STSTAT register by SCU.

- HWCFG[6] and HWCFG[1,2] are recognized as high when the respective pin is open or pulled up to VEXT supply with pull device  $> 2\text{ k}\Omega$  and  $< 4.7\text{ k}\Omega$  on the external system.
- HWCFG[6] and HWCFG[1,2] are recognized as low when the respective pin is pulled down to GND with pull device  $> 2\text{ k}\Omega$  and  $< 4.7\text{ k}\Omega$  on the external system.

The lower limit of the pull resistance is derived from the overload and short specification (see data sheet) in case of a short event.

Regardless of the HWCFG[6] setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate after standby mode entry.

## Power Management System for Low-End (PMSLE)



**Figure 122 Hardware Configuration (HWCFG) pins**

**Table 381 Supply Mode and Topology selection**

No.	HWCFG [2,1] <sup>1)</sup>	VCAPO VCAP1 <sup>2)</sup>	Supply Pin Voltage Level / Source <sup>3)</sup>	Selected Supply Scheme
a.)	11 <sub>B</sub>	VCAPO/VCAP1 pins connected to external flying capacitor.	VEXT & VEVRSB = 5.0 V external supply. VDDM = VAREFx = 5V or 3.3V external supply. VFLEX = 5 V or 3.3 V. VDDP3 and VDDFL3 supplied by EVR33. VDD supplied by EVRC.	5 V single source supply. EVRC in SMPS mode. EVR33 in LDO mode. 5 V or 3.3 V ADC domain. 5 V or 3.3 V Flexport domain. Standby Mode supported.
d.)	01 <sub>B</sub>	VCAPO/VCAP1 pins can be left open.	VEXT & VEVRSB = 5.0 V external supply. VDDM = VAREFx = 5V or 3.3V external supply. VFLEX = 5 V or 3.3 V. VDDP3 and VDDFL3 supplied by EVR33. VDD = 1.25 V external supply.	5 V & 1.25 V external supply. EVRC inactive. EVR33 in LDO mode. 5 V or 3.3 V ADC domain. 5 V or 3.3 V Flexport domain. Standby Mode is supported and 1.25V supply shall be switched off by external regulator after Standby state is entered.

**Power Management System for Low-End (PMSLE)**
**Table 381 Supply Mode and Topology selection (cont'd)**

No.	HWCFG [2,1] <sup>1)</sup>	VCAP0 VCAP1 <sup>2)</sup>	Supply Pin Voltage Level / Source <sup>3)</sup>	Selected Supply Scheme
e.)	10 <sub>B</sub>	VCAP0/VCAP1 pins connected to external flying capacitor.	VEXT, VEVRSB, VDDP3, VFLEX and VDDFL3 = 3.3V external supply. VDDM = VAREFx = 5V or 3.3V external supply. VDD supplied by EVRC.	3.3 V single source supply. EVRC in SMPS mode. EVR33 inactive. 5 V or 3.3 V ADC domain. 3.3 V Flexport domain. Standby Mode supported.
			VEXT & VEVRSB = 5.0 V external supply. VDDP3, VFLEX and VDDFL3 = 3.3V external supply. VDDM = VAREFx = 5V or 3.3V external supply. VDD supplied by EVRC.	5 V & 3.3 V external supply. EVRC in SMPS mode. EVR33 inactive. 5 V or 3.3 V ADC domain. 5 V or 3.3 V Flexport domain. Standby Mode is supported and 3.3V supply shall be switched off by external regulator after Standby state is entered.
h.)	00 <sub>B</sub>	VCAP0/VCAP1 pins can be left open.	VEXT & VEVRSB = 5.0 V external supply. VDDM = VAREFx = 5V or 3.3V external supply. VFLEX = 5 V or 3.3 V external supply. VDDP3 and VDDFL3 = 3.3V external supply. VDD = 1.25 V external supply.	5 V, 3.3 V and 1.25 V are supplied externally. EVRC and EVR33 inactive. 5 V or 3.3 V ADC domain. 5 V or 3.3 V Flexport domain. Standby Mode is supported and 3.3V and 1.25V supplies shall be switched off by external regulator after Standby state is entered.

- 1) if HWCFG[2,1] pins are left unconnected, it is ensured that EVR33 and EVRC are active owing to the internal weak pull-up active by default after start-up/cold PORST.
- 2) VCAPx pins are dedicated for SMPS Switch capacitor regulator and cannot be used as port pins, In SMPS mode, the VCAPx shall be connected with a flying capacitor between the pins. In case EVRC is disabled, the VCAP pins may be left open or may be connected with a small decoupling capacitor. VCAP pins shall not be connected to ground as ESD diodes start conducting.
- 3) Only Nominal supply voltage values of respective rails are indicated in the table. The tolerances of the supply voltages are documented in datasheet.

**Power Management System for Low-End (PMSLE)**
**Table 382 5 V Nominal Supply : Voltage variations at independent supply rails during system modes**

<b>Voltage Rail</b>	<b>5 V Start-up till cold PORST release</b>	<b>5 V Operation RUN mode SLEEP mode</b>	<b>5 V Cranking</b>	<b>5 V VEVRSB STANDBY mode</b>	<b>5 V (VEVRSB + VEXT) STANDBY mode</b>	<b>5 V ED STANDBY mode</b>
$V_{EVRSB}$	2.6 - 5.5 EVRx Start-up	4.5 - 5.5	2.97 - 5.5	2.6 - 5.5	2.97 - 5.5	0 V
$V_{EXT}$	2.6 - 5.5 EVRx Start-up	4.5 - 5.5	2.97 - 5.5	0 V	2.97 - 5.5	
$V_{FLEX}$	Supplied modules in reset	2.97 - 3.63 4.5 - 5.5	2.97 - 5.5		2.97 - 5.5 0 V	
$V_{DDM}$		2.97 - 3.8 3.8 - 5.5	2.97 - 5.5		2.97 - 5.5 0 V	
$V_{DDP3}$	Supply Ramp-up	2.97 - 3.63	2.6 - 3.63 <sup>1)</sup>		0 V	
$V_{DD}$	Phase. Supplied modules in reset	1.125 - 1.375	1.125 - 1.375			1.0 <sup>2)</sup> - 1.375
$V_{DDSB}$						
$V_{DDPD}$ <sup>3)</sup>	1.15 - 1.43	1.15 - 1.43	1.15 - 1.43	1.15 - 1.43	1.15 - 1.43	0 V

1) If EVR33 is used, a minimum VEXT voltage is required to account for pass device drop as documented in datasheet PMS EVR33 section. The voltage is allowed to drop to 2.6V after Flash is set cranking mode where only reading from Flash is allowed with increased wait states.

2) 1.0 V permitted at VDDSB only for ED RAM data retention mode as documented in Emulation device section.

3) Supply level at internal VDDPD pad

**Table 383 3.3 V Nominal Supply : Voltage variations at independent supply rails during system modes**

<b>Voltage Rail</b>	<b>3.3 V Start-up till cold PORST release</b>	<b>3.3 V Operation RUN mode SLEEP mode</b>	<b>3.3 V Cranking</b>	<b>3.3 V VEVRSB STANDBY mode</b>	<b>3.3 V (VEVRSB + VEXT) STANDBY mode</b>	<b>3.3 V ED STANDBY mode</b>
$V_{EVRSB}$	2.6 - 3.63 EVRx Start-up	2.97 - 3.63	2.97 - 3.63	2.6 - 3.63	2.97 - 3.63	0 V
$V_{EXT}$	2.6 - 3.63 EVRx Start-up	2.97 - 3.63	2.97 - 3.63	0 V	2.97 - 3.63	
$V_{FLEX}$	Supplied modules in reset	2.97 - 3.63	2.97 - 3.63		2.97 - 3.63 0 V	
$V_{DDM}$		2.97 - 3.8 3.8 - 5.5	2.97 - 5.5		2.97 - 3.63 0 V	
$V_{DDP3}$	Supply Ramp-up	2.97 - 3.63	2.97 - 3.63		0 V	
$V_{DD}$	Phase. Supplied modules in reset	1.125 - 1.375	1.125 - 1.375			1.0 - 1.375
$V_{DDSB}$						
$V_{DDPD}$	1.15 - 1.43	1.15 - 1.43	1.15 - 1.43	1.15 - 1.43	1.15 - 1.43	0 V

## Power Management System for Low-End (PMSLE)

**Table 384 Allowed Combinations of Nominal External Supply Voltages between Voltage Rails<sup>1)</sup>**

Supply Rails	<b>VEXT = VEVRSB = 5V Nominal Voltage Level</b>				<b>VEXT = VEVRSB = 3.3V Nominal Voltage Level</b>	
$V_{EVRSB}$	5 V <sup>2)</sup>				3.3 V	
$V_{EXT}$	5 V				3.3 V	
$V_{FLEX}$ <sup>3)</sup>	5 V	3.3 V	5 V	3.3 V	3.3 V	
$V_{DDM}$ <sup>4)</sup>	5 V				5 V <sup>5)</sup>	3.3 V
$V_{DDP3}$ <sup>6)</sup>	3.3 V (external supply or generated by EVR33 )				3.3 V (external supply or generated by EVR33 )	
$V_{DD}$	1.25 V (external supply or generated by EVRC )				1.25 V (external supply or generated by EVRC )	
$V_{DDSB}$ <sup>7)</sup>	1.25 V (external supply or generated by EVRC )				1.25 V (external supply or generated by EVRC )	

- 1) All supply rails shall have ramped up to their minimum voltage operational limits as documented in the datasheet before warm PORST reset release. It is not allowed to leave any supply rail un supplied after warm PORST reset release.
- 2) VEVRSB supply rail can be ramped down during VEVRSB Standby mode to 2.6 V minimum voltage.
- 3) VFLEX supply rail provides supply to ports P11 and P12 and can be supplied with nominal 3.3V supply when remaining ports are supplied with nominal 5V. VFLEX maybe supplied by the same external supply source connected also to VEXT supply rail. VFLEX supply level shall be less than or equal to VEXT supply level.
- 4) VDDM analog supply and VAREFx analog reference supply shall have the same supply level. It is recommended to supply VDDM and VAREFx from the same external supply source with filters.
- 5) VDDM supplies only a part of analog pins. For shared analog pins supplied by VEXT (P00) and VEVRSB (P33), the voltage levels of the resepective analog channels would be bounded by the respective supply voltages when they are lower than the VDDM / VAREF voltages.
- 6) EVR33 is designed to supply the current required only by VDDP3 rail and the associated modules requiring 3.3V supply. It is not intended to supply VFLEX pad currents from 3.3V VDDP3 rail when EVR33 generates VDDP3 supply.
- 7) VDDSB shall be connected to VDD rail and supplied together in case of non emulation devices.

## Power Management System for Low-End (PMSLE)

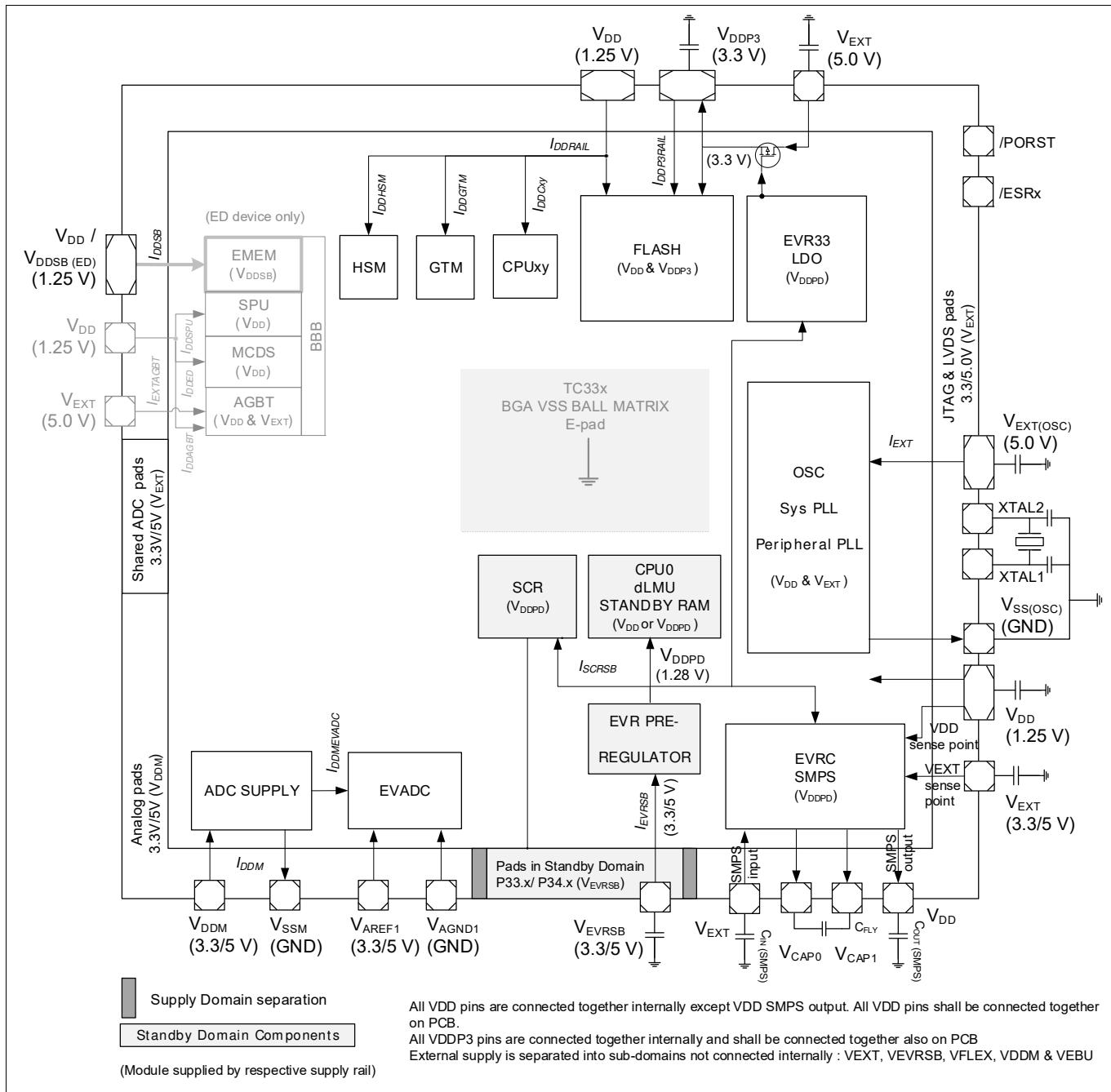


Figure 123 TC33x Supply Pins and Module Connectivity

## Power Management System for Low-End (PMSLE)

### 12.2.1.2 Supply Ramp-up and Ramp-down Behavior

#### 12.2.1.2.1 Single Supply mode (a)

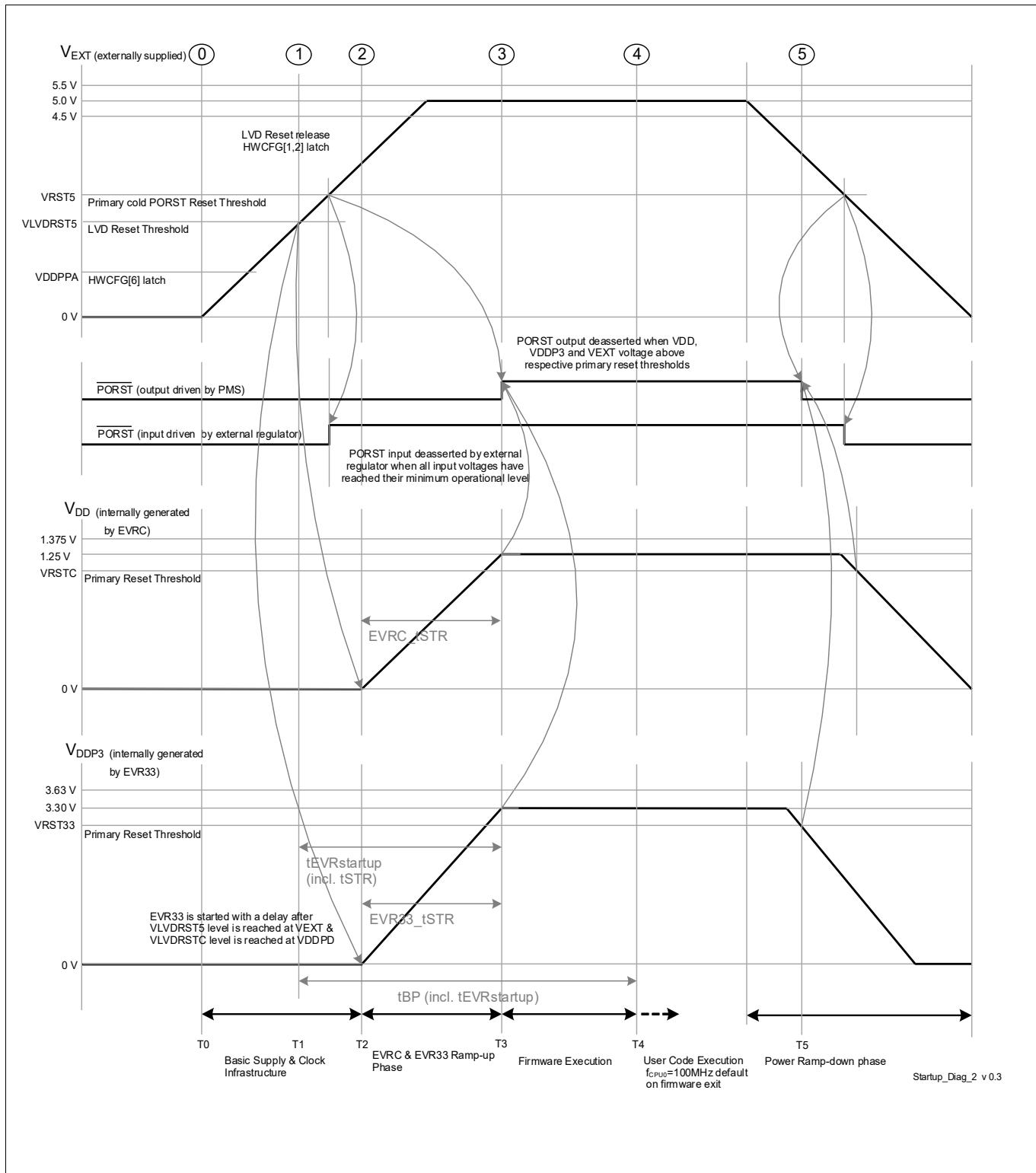


Figure 124 Single Supply mode (a) - VEXT (5 V) single supply

## Power Management System for Low-End (PMSLE)

VEXT = 5 V single supply mode. VDD and VDDP3 are generated internally by the EVRC and EVR33 internal regulators.

- The rate at which current is drawn from the external regulator ( $dI_{EXT}/dt$ ) is limited during the basic infrastructure and EVRx regulator start-up phase (T0 up to T3) to a maximum of 100 mA with 50  $\mu$ s settling time. Start-up slew rates for supply rails shall comply to datasheet parameter SR. The slope is defined as the maximal tangential slope between 0% to 100% voltage level. Actual waveform may not represent the specification.
- Furthermore it is also ensured that the current drawn from the regulator ( $dI_{DD}/dt$ ) is limited during the Firmware start-up phase (T3 up to T4) to a maximum of 100 mA with 100  $\mu$ s settling time.
- PORST is active/asserted when either PORST (input) or PORST (output) is active/asserted.
- PORST (input) active means that the reset is held active by external agents by pulling the PORST pin low. It is recommended to keep the PORST (input) asserted until the external supply is above the respective primary reset threshold.
- PORST (output) active means that  $\mu$ C asserts the reset internally and drives the PORST pin low thus propagating the reset to external devices. The PORST (output) is asserted by the  $\mu$ C when atleast one among the three supply domains (VDD, VDDP3 or VEXT) violate their primary under-voltage reset thresholds. The PORST (output) is de-asserted by the  $\mu$ C when all supplies are above their primary reset thresholds and the basic supply and clock infrastructure is available. During reset release at T3, the load jump of upto 150 mA ( $dI_{DD}$ ) is expected.
- The power sequence as shown in [Figure 124](#) is enumerated below
  - T1 up to T2 refers to the period in time when basic supply and clock infrastructure components are available as the external supply ramps up. The bandgap and internal clock sources are started .The supply mode is evaluated based on the HWCFG[2:1,6] pins. These events are initiated after LVD reset release at T1. LVD reset is released when both input voltages VEXT and VEVRSB are above VLVRST5 and VLVRSTSB levels respectively. Internal pre-regulator VDDPD voltage is above VLVRSTC level.
  - T2 refers to the point in time where consequently a soft start of EVRC and EVR33 regulators are initiated. PORST (input) does not have any affect on EVR33 or EVRC output and regulators continue to generate the respective voltages though PORST is asserted and the device is in reset state. The generated voltage follows a soft ramp-up over the tSTR (datasheet parameter) time to avoid overshoots.
  - T3 refers to the point in time when all supplies are above their primary reset thresholds denoted by VRST5, VRST33 and VRSTC supply voltage levels. EVRC and EVR33 regulators have ramped up. PORST (output) is de-asserted and HWCFG[3:5] pins are latched on PORST rising edge by SCU. Firmware execution is initiated. The time between T1 and T3 is documented as tEVRstartup (datasheet parameter).
  - T4 refers to the point in time when Firmware execution is completed and User code execution starts with CPU0 at a default frequency of 100 MHz. The time between T0 and T4 is documented as tBP (datasheet parameter).
  - T5 refers to the point in time during the ramp-down phase when at least one of the externally provided or generated supplies (VDD, VDDP3 or VEXT) drop below their respective primary under-voltage reset thresholds.

## Power Management System for Low-End (PMSLE)

### 12.2.1.2.2 Single Supply mode (e)

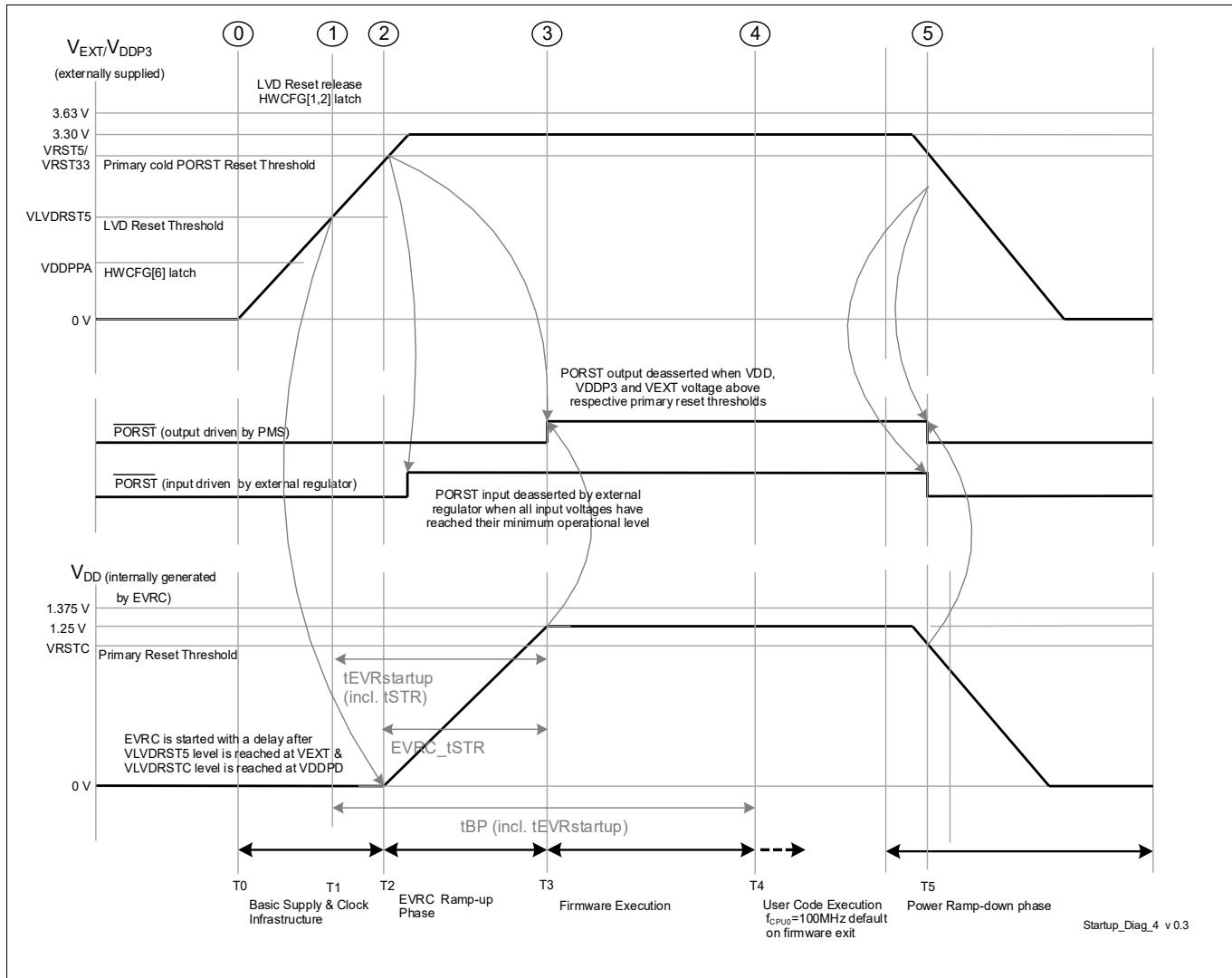


Figure 125 Single Supply mode (e) - ( $V_{EXT}$  &  $V_{DDP3}$ ) 3.3 V single supply

$V_{EXT} = V_{DDP3} = 3.3$  V single supply mode.  $V_{DD}$  is generated internally by the EVRC regulator.

- The rate at which current is drawn from the external regulator ( $dI_{EXT} / dt$ ) is limited in the Start-up phase (T2 up to T3) to a maximum of 100 mA with 50  $\mu$ s settling time. Start-up slew rates for supply rails shall comply to datasheet parameter SR. The slope is defined as the maximal tangential slope between 0% to 100% voltage level. Actual waveform may not represent the specification.
- PORST is active/asserted when either PORST (input) or PORST (output) is active/asserted.
- PORST (input) active means that the reset is held active by external agents by pulling the PORST pin low. It is recommended to keep the PORST (input) asserted until the external supply is above the respective primary reset threshold.
- PORST (output) active means that  $\mu$ C asserts the reset internally and drives the PORST pin low thus propagating the reset to external devices. The PORST (output) is asserted by the  $\mu$ C when at least one among the three supply domains ( $V_{DD}$ ,  $V_{DDP3}$  or  $V_{EXT}$ ) violate their primary under-voltage reset thresholds. The PORST (output) is de-asserted by the  $\mu$ C when all supplies are above their primary reset thresholds and the

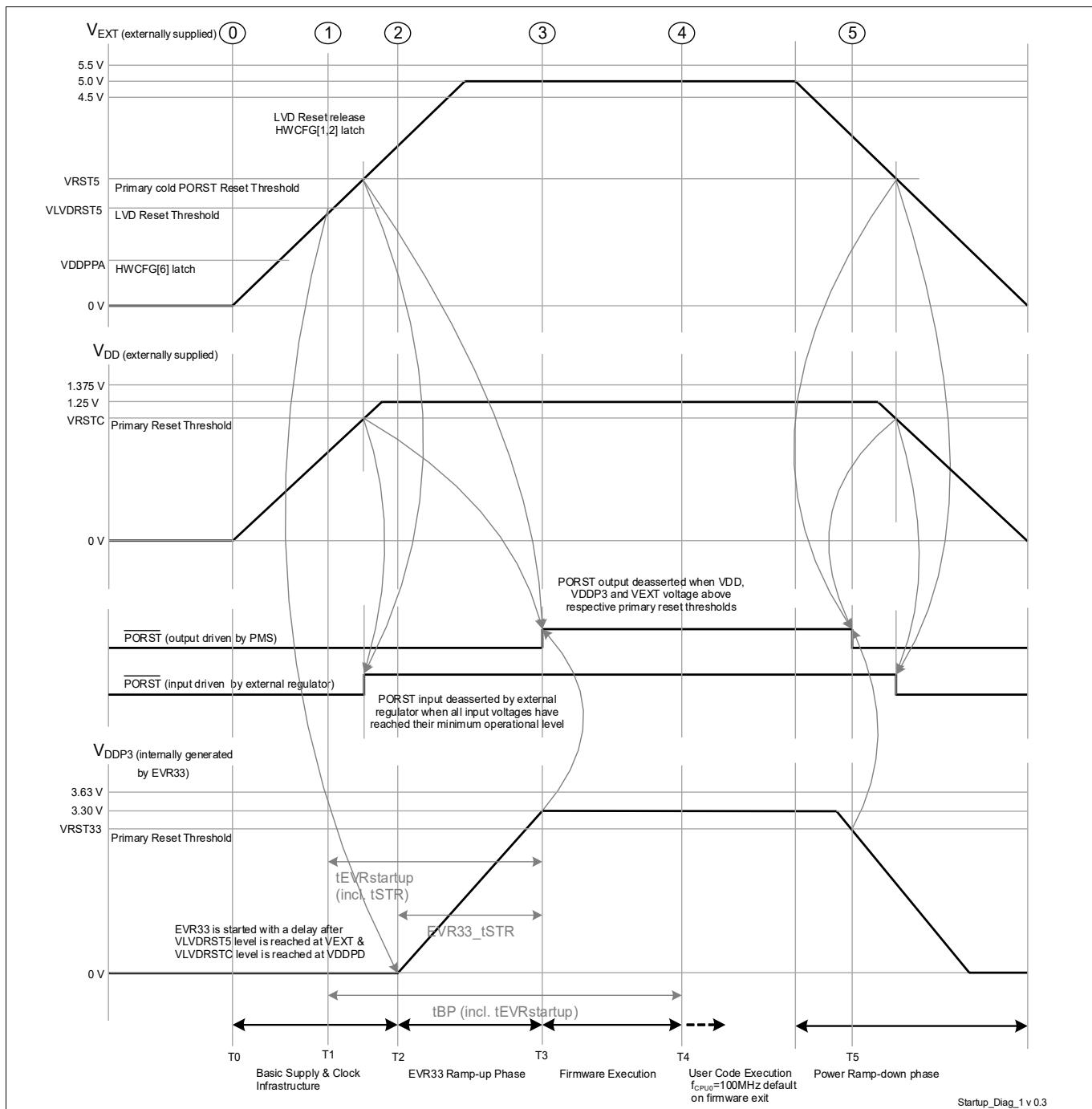
## Power Management System for Low-End (PMSLE)

basic supply and clock infrastructure is available. During reset release at T3, the load jump of upto 150 mA (dIDD) is expected.

- The power sequence as shown in [Figure 125](#) is enumerated below
  - T1 up to T2 refers to the period in time when basic supply and clock infrastructure components are available as the external supply ramps up. The bandgap and internal clock sources are started .The supply mode is evaluated based on the HWCFG[2:1,6] pins. These events are initiated after LVD reset release at T1. LVD reset is released when both input voltages VEXT and VEVRSB are above VLDRST5 and VLDRSTSB levels respectively. Internal pre-regulator VDDPD voltage is above VLDRSTC level.
  - T2 refers to the point in time where consequently a soft start of EVRC regulator is initiated. PORST (input) does not have any affect on EVRC output and regulators continue to generate the respective voltages though PORST is asserted and the device is in reset state. The generated voltage follows a soft ramp-up over the tSTR (datasheet parameter) time to avoid overshoots.
  - T3 refers to the point in time when all supplies are above their primary reset thresholds denoted by VRST5, VRST33 and VRSTC supply voltage levels. EVRC regulator has ramped up. PORST (output) is de-asserted and HWCFG[3:5] pins are latched on PORST rising edge by SCU. Firmware execution is initiated. The time between T1 and T3 is documented as tEVStartup (datasheet parameter).
  - T4 refers to the point in time when Firmware execution is completed and User code execution starts with CPU0 at a default frequency of 100 MHz. The time between T0 and T4 is documented as tBP (datasheet parameter).
  - T5 refers to the point in time during the ramp-down phase when at least one of the externally provided or generated supplies (VDD, VDDP3 or VEXT) drop below their respective primary under-voltage reset thresholds.

## Power Management System for Low-End (PMSLE)

### 12.2.1.2.3 External Supply mode (d)



**Figure 126 External Supply mode (d) - VEXT and VDD externally supplied**

VEXT = 5 V and VDD supplies are externally supplied. 3.3V is generated internally by the EVR33 regulator.

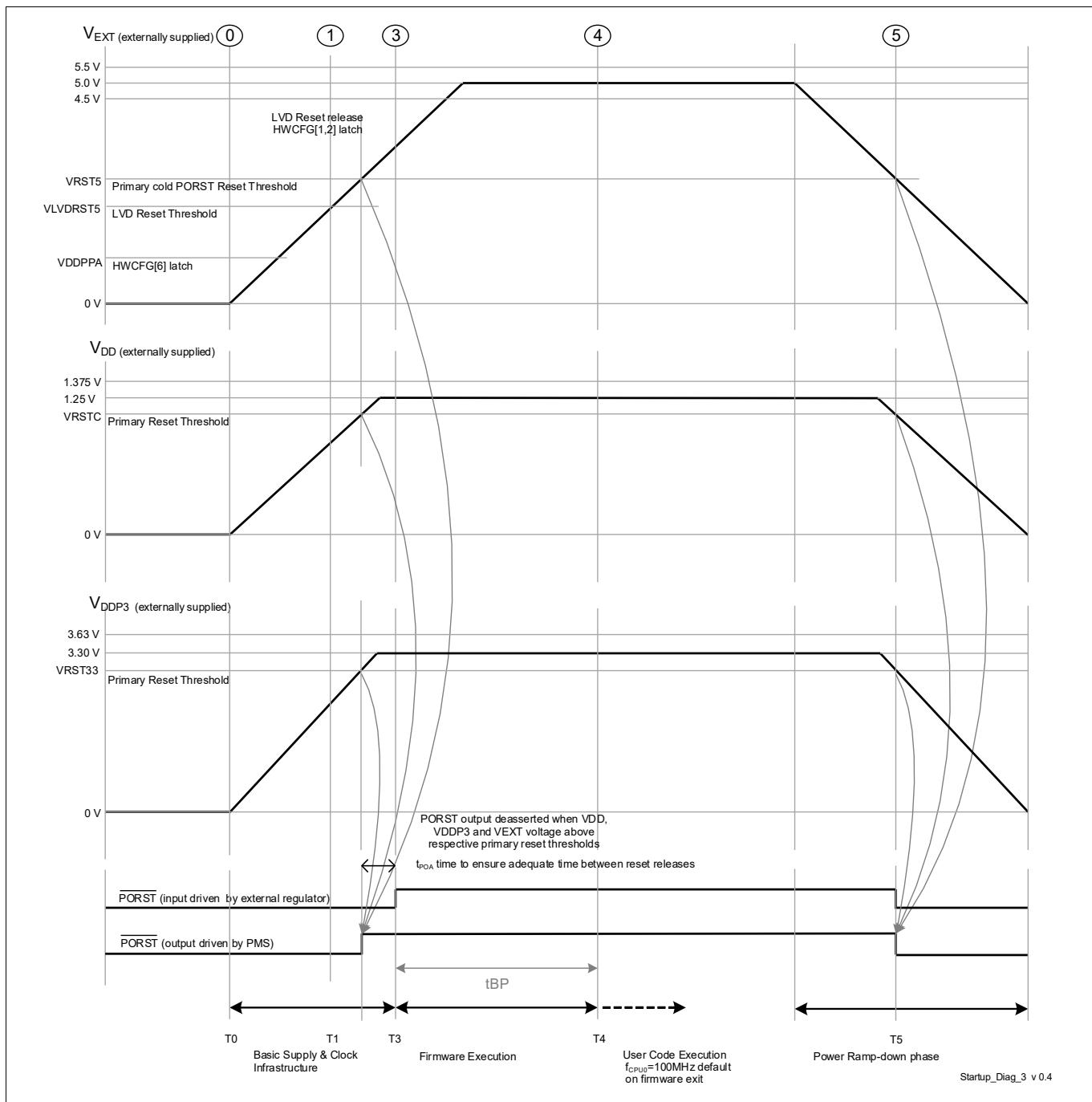
- External supplies VEXT and VDD may ramp-up or ramp-down independent of each other with regards to start, rise and fall time(s). Start-up slew rates for supply rails shall comply to datasheet parameter SR. The slope is defined as the maximal tangential slope between 0% to 100% voltage level. Actual waveform may not represent the specification. It is expected that during start-up, VEXT ramps up before VDD rail. In case VDD voltage rail is ramped up before VEXT; VDD supply overshoots during start-up shall be limited within the operational voltage range.

## Power Management System for Low-End (PMSLE)

- The rate at which current is drawn from the external regulator ( $dI_{EXT}/dt$  or  $dI_{DD}/dt$ ) is limited in the Start-up phase to a maximum of 100 mA with 100  $\mu$ s settling time.
- PORST is active/asserted when either PORST (input) or PORST (output) is active/asserted.
- PORST (input) active means that the reset is held active by external agents by pulling the PORST pin low. It is recommended to keep the PORST (input) asserted until all the external supplies are above their primary reset thresholds.
- PORST (output) active means that  $\mu$ C asserts the reset internally and drives the PORST pin low thus propagating the reset to external devices. The PORST (output) is asserted by the  $\mu$ C when at least one among the three supply domains (VDD, VDDP3 or VEXT) violate their primary under-voltage reset thresholds. The PORST (output) is de-asserted by the  $\mu$ C when all supplies are above their primary reset thresholds and the basic supply and clock infrastructure is available. During reset release at T3, the load jump of up to 150 mA ( $dI_{DD}$ ) is expected.
- The power sequence as shown in [Figure 126](#) is enumerated below
  - T1 up to T2 refers to the period in time when basic supply and clock infrastructure components are available as the external supply ramps up. The bandgap and internal clock sources are started. The supply mode is evaluated based on the HWCFG[2:1,6] pins. These events are initiated after LVD reset release at T1. LVD reset is released when both input voltages VEXT and VEVRSB are above VLVRST5 and VLVRSTSB levels respectively. Internal pre-regulator VDDPD voltage is above VLVRSTC level.
  - T2 refers to the point in time where consequently a soft start of EVR33 regulator is initiated. PORST (input) does not have any affect on EVR33 output and regulators continue to generate the respective voltages though PORST is asserted and the device is in reset state. The generated voltage follows a soft ramp-up over the tSTR (datasheet parameter) time to avoid overshoots.
  - T3 refers to the point in time when all supplies are above their primary reset thresholds denoted by VRST5, VRST33 and VRSTC supply voltage levels. EVR33 regulators has ramped up. PORST (output) is de-asserted and HWCFG[3:5] pins are latched on PORST rising edge by SCU. Firmware execution is initiated. The time between T1 and T3 is documented as tEVStartup (datasheet parameter).
  - T4 refers to the point in time when Firmware execution is completed and User code execution starts with CPU0 at a default frequency of 100 MHz. The time between T0 and T4 is documented as tBP (datasheet parameter).
  - T5 refers to the point in time during the ramp-down phase when at least one of the externally provided or generated supplies (VDD, VDDP3 or VEXT) drop below their respective primary under-voltage reset thresholds.

## Power Management System for Low-End (PMSLE)

### 12.2.1.2.4 External Supply mode (h)



**Figure 127 External Supply mode (h) - VEXT, VDDP3 & VDD externally supplied**

All supplies, namely VEXT, VDDP3 & VDD are externally supplied.

- External supplies VEXT, VDDP3 & VDD may ramp-up or ramp-down independent of each other with regards to start, rise and fall time(s). Start-up slew rates for supply rails shall comply to datasheet parameter SR. The slope is defined as the maximal tangential slope between 0% to 100% voltage level. Actual waveform may not represent the specification. It is expected that during start-up, VEXT ramps up before VDDP3 and VDD rails. If smaller voltage rails are ramped up before VEXT; VDD and VDDP3 supply overshoots during start-up shall be limited within the operational voltage ranges of the respective rails.

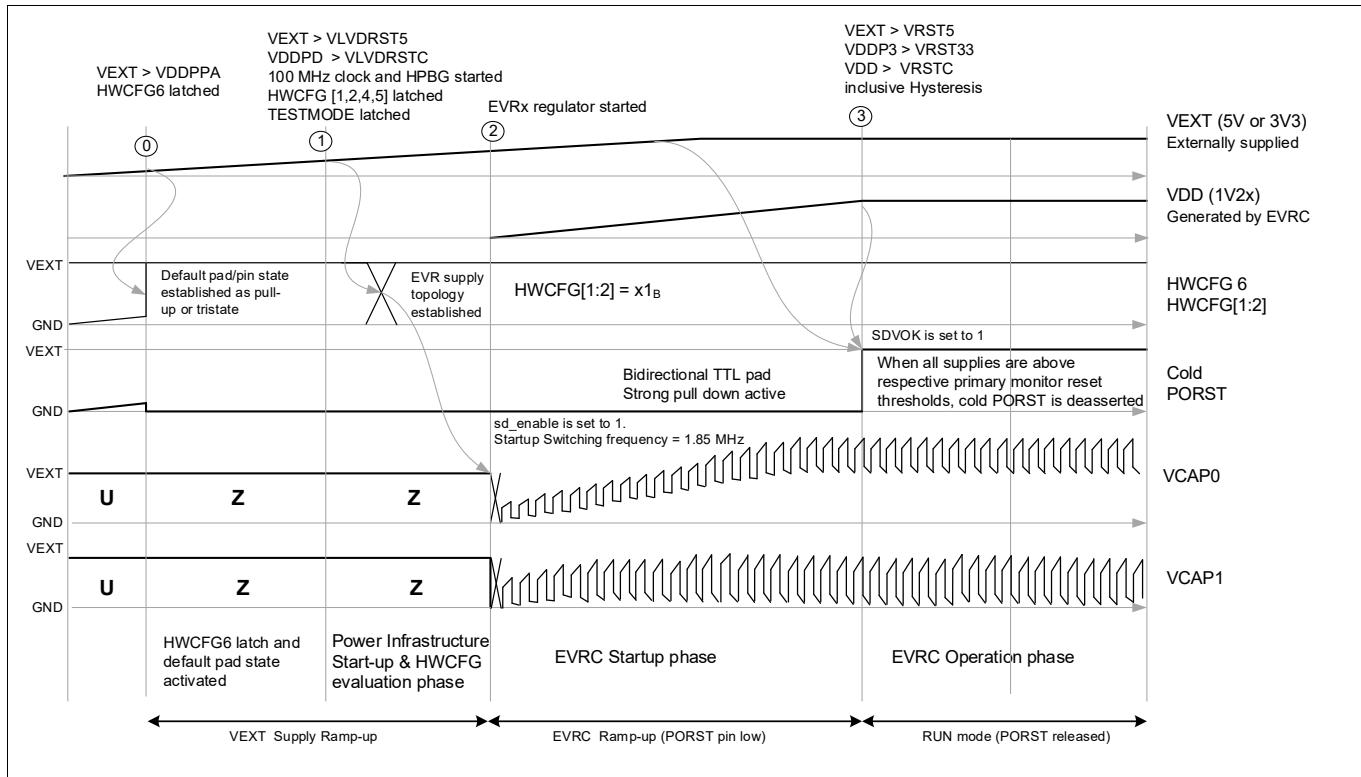
## Power Management System for Low-End (PMSLE)

- The rate at which current is drawn from the external regulator ( $dI_{EXT}/dt$ ,  $dI_{DD}/dt$  or  $dI_{DDP3}/dt$ ) is limited in the Start-up phase to a maximum of 100 mA with 50  $\mu$ s settling time.
- PORST is active/asserted when either PORST (input) or PORST (output) is active/asserted.
- PORST (input) active means that the reset is held active by external agents by pulling the PORST pin low. It is recommended to keep the PORST (input) asserted until all the external supplies are above their primary reset thresholds.
- PORST (output) active means that  $\mu$ C asserts the reset internally and drives the PORST pin low thus propagating the reset to external devices. The PORST (output) is asserted by the  $\mu$ C when at least one among the three supply domains (VDD, VDDP3 or VEXT) violate their primary under-voltage reset thresholds. The PORST (output) is de-asserted by the  $\mu$ C when all supplies are above their primary reset thresholds and the basic supply and clock infrastructure is available. During reset release at T3, the load jump of up to 150 mA ( $dI_{DD}$ ) is expected.
- The power sequence as shown in [Figure 127](#) is enumerated below
  - T1 up to T3 refers to the period in time when basic supply and clock infrastructure components are available as the external supply ramps up. The bandgap and internal clock sources are started. The supply mode is evaluated based on the HWCFG[2:1,6] pins. These events are initiated after LVD reset release at T1. LVD reset is released when both input voltages VEXT and VEVRSB are above VLVRST5 and VLVRSTSB levels respectively. Internal pre-regulator VDDPD voltage is above VLVRSTC level.
  - T3 refers to the point in time when all supplies are above their primary reset thresholds denoted by VRST5, VRST33 and VRSTC supply voltage levels. PORST (output) is de-asserted and HWCFG[3:5] pins are latched on PORST rising edge by SCU. Firmware execution is initiated.
  - T4 refers to the point in time when Firmware execution is completed and User code execution starts with CPU0 at a default frequency of 100 MHz. The time between T0 and T4 is documented as tBP (datasheet parameter).
  - T5 refers to the point in time during the ramp-down phase when at least one of the externally provided supplies (VDD, VDDP3 or VEXT) drop below their respective primary under-voltage reset thresholds.

## Power Management System for Low-End (PMSLE)

### 12.2.1.2.5 EVRC, VCAPx behavior during Start-up

If EVRC regulator is activated via HWCFG[2] = 1, then the behavior of the pins during start-up is as portrayed in **Figure 128**. During VEXT ramp-up, the VCAPx pins are in tristate. Once the HWCFG[2:1] pins are latched and the internal enable signal for the SCDCDC is activated (sd\_enable = 1), EVRC is starting up and the VCAPx pins are driven by the internal switch network at the start-up switching frequency of 1.85 MHz. During the start-up phase, the flying capacitor is charged to the target output voltage and a soft start-up procedure is applied to avoid current overshoots, as described in **Section 12.2.2.2**.



**Figure 128** VCAP behavior during start-up when EVRC regulator is used

## Power Management System for Low-End (PMSLE)

### 12.2.1.3 PMS Infrastructure Components

Power Management System constitutes infrastructure components which need to be started before ramping the EVR33 & EVRC Embedded Voltage Regulators.

- EVR Pre- Regulator (EVRPR)
- 100 MHz Back-up Clock Source (fBACK)
- Secondary High Precision Bandgap reference (SHPBG)
- 70 kHz Standby Clock Source (fSB)
- Primary Low Power Bandgap reference (PLPBG)

#### 12.2.1.3.1 Independent VEVRSB & VDDPD Supply domain and EVR Pre-Regulator (EVRPR)

The objective of the EVRPR is to supply the basic infrastructure components, the Standby domain and certain safety components with a dedicated low-noise independent supply. The EVRPR pre-regulator is supplied directly by the external 5 V or 3.3 V VEVRSB supply. It is implemented as a low drop-out regulator generating the 1.25 V VDDPD internal voltage which is buffered internally and is not routed to any external supply pin. Since EVRPR part is always powered on as long as the external supply is available and also in Standby mode, it is implemented to have low power consumption to meet ISTANDBY current parameter limits in datasheet. The EVRPR supplies the high precision bandgap, the 100 MHz EVR clock source and EVRC / EVR33 regulators as the regulators have to be independent from their generated supplies. The EVRPR also supplies the Standby domain including the Standby RAMs, the Wake-Up Timer, the Standby Controller and a part of the Port domain (Port 33 / 34).

The minimum power detection logic ensures that a minimum voltage level is available on VEXT and VEVRSB external supplies and on the internally generated VDDPD supply via dedicated detectors. The VEXT supply is monitored for minimum VLDRST5 level to ensure that adequate voltage is available to latch HWCFG pins and start EVRC. Likewise, the internal VDDPD supply is monitored for minimum VLDRSTC voltage level by the VDDPD detector with in-built reference. When both conditions are fulfilled the start-up of the EVRPR has been successfully completed and the EVR Low Voltage Detector reset (LVD reset) is released. The 100 MHz clock and high precision bandgap are consequently started. The HWCFG pins are evaluated to establish the supply mode which needs to be activated. Consequently EVRC and EVR33 are started in parallel in a soft ramp-up to ensure a voltage ramp-up with minimal overshoots. In case both EVRC and EVR33 are activated, a normal start-up is completed when both the regulator outputs are stable and operational. Consequently cold PORST reset is released when VEXT, VDDP3 and VDD voltages ramp-ups are complete and the respective voltages are above their minimum operational limits (VRSTxx / VxxPRIUV).

#### 12.2.1.3.2 Reference Voltage Generation : Secondary Bandgap Reference (SHPBG)

The objective of the Secondary High Precision BandGap and Reference current circuitry is to provide an accurate voltage reference and reference currents to various modules. The reference is used by EVRC, EVR33, supply monitors, ADC modules, XTAL Oscillator, Flash, ADC and LVDS Pads.

The secondary high precision bandgap reference is checked against the primary low power bandgap reference or VDDPD voltage to detect bandgap drifts during start-up phase. This is part of the Power BIST ([Section 12.2.2.5.3](#)) which is carried out only during a supply ramp-up.

#### 12.2.1.3.3 100 MHz Back-up Clock Source (fBACK)

The 100 MHz clock source is a precise back-up on-chip clock used by EVRs, firmware and serves as the main system clock during the Start-up phase. It is further used as an independent clock reference for clock monitoring and can be used as a back-up clock in case of loss of lock or crystal failures. After start-up, the 100 MHz clock source has a higher variance in the order of  $\pm 30\%$  and the clock source is later trimmed by the start-up software

---

**Power Management System for Low-End (PMSLE)**

as documented in datasheet. It shall be ensured that the PMS subsystem, boot software / Firmware and the start-up modules are tolerant and functionally robust to this clock variation.

The **EVROSCCTRL** register shall not be modified by the application software, as it is configured by the Start-Up Software in order to trim the back-up oscillator to the specified accuracy limits. Additional compensation for improved accuracy across the temperature range is possible by enabling the dynamic oscillator trimming in the register bits **EVROSCCTRL.OSCTEMPOFFS** and **EVROSCCTRL.OSCTRIMEN**.

---

**Power Management System for Low-End (PMSLE)**

#### 12.2.1.4 Die Temperature Measurement

The Die Temperature Sensor (DTS) generates a measurement result that indicates directly the current temperature. The DTS measures the temperature with an accuracy within ( $T_{NL} + T_{CALACC}$ ) parameter limits within the TSR temperature range documented in the datasheet. The result of the measurement is updated periodically in **DTSSTAT.RESULT** register bit field with a resolution less than 1/5th of a degree Kelvin. The Die Temperature Sensor is available after cold PORST reset release on a device start-up and temperature measurements are carried out continuously during normal RUN / SLEEP modes. The DTS and corresponding registers are not affected by a warm PORST, system or application reset; consequently DTSTAT temperature result from earlier conversion is available for immediate use after any warm reset.

After an ongoing temperature measurement is completed, **DTSSTAT.RESULT** bit field is updated coherently with the new value. An interrupt service request (SRC\_PMSDTS) can be generated after a measurement is completed. The DTS accuracy and measurement time is defined in the Data Sheet.

Die temperature upper and lower limits are configured in **DTSLIM.UPPER** and **LOWER** register bits. On violation of these limits, **DTSLIM.UOF** and **LLU** status bits are set and alarms are forwarded to SMU and HSM. After start-up, the DTS limits have to be re-configured appropriately depending on the application before alarm reactions from SMU or HSM are activated. Only when a new DTS conversion result is available, the DTS comparators are consequently triggered to check the actual **DTSSTAT.RESULT** against the upper and lower limits.

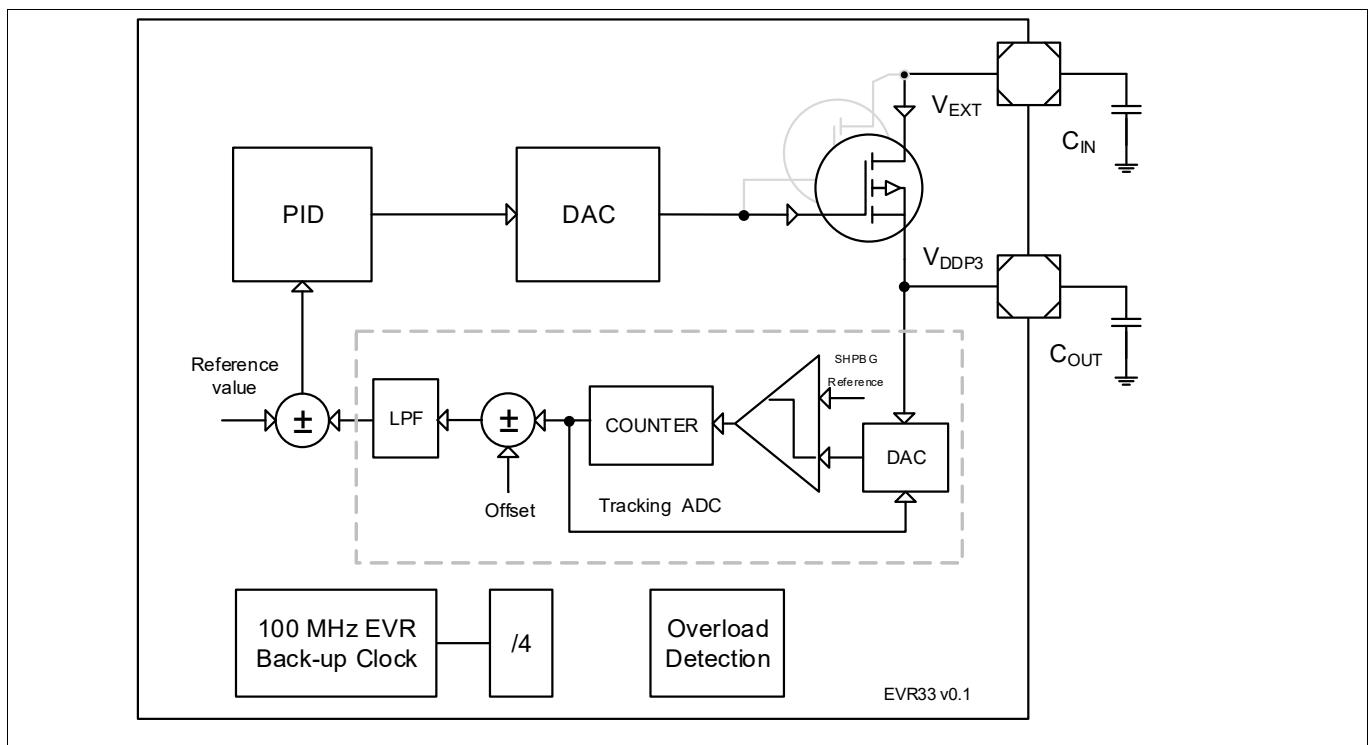
Note: LDMST or SWAPMSK.W should be used only with bit mask enabled for all 'rwh' bits in the **DTSLIM** register.

## Power Management System for Low-End (PMSLE)

### 12.2.2 Power Supply Generation and Monitoring

#### 12.2.2.1 Linear Regulator Mode (EVR33)

The EVR33 regulator supplies the Flash module. EVR33 constitutes a digital regulator, a pass device control unit and a voltage feedback loop. In order to compensate technology and process variations, the ADC and the DAC are device individually trimmed. The EVR33 regulator output voltage ( $V_{DDP3}$ ) is measured by a dedicated ADC using SHPBG reference supply and the result is indicated also in register **EVRADCSTAT.ADC33V**. The closed loop regulation cycle is triggered at the end of the ADC conversion. The error difference is fed to a PID controller and the output of the controller is fed to the DAC to control the gate voltage of the pass devices. The pass device outputs are buffered by external capacitor to handle load transients so as not to violate the operating voltage limits. EVR33 can be individually disabled via HWCFG[1] pin as described in [Chapter 12.2.1.1](#). During the Start-up phase, the setpoint voltage is ramped up in steps over the start-up period to ensure a soft ramp-up of the  $V_{DDP3}$  voltage.

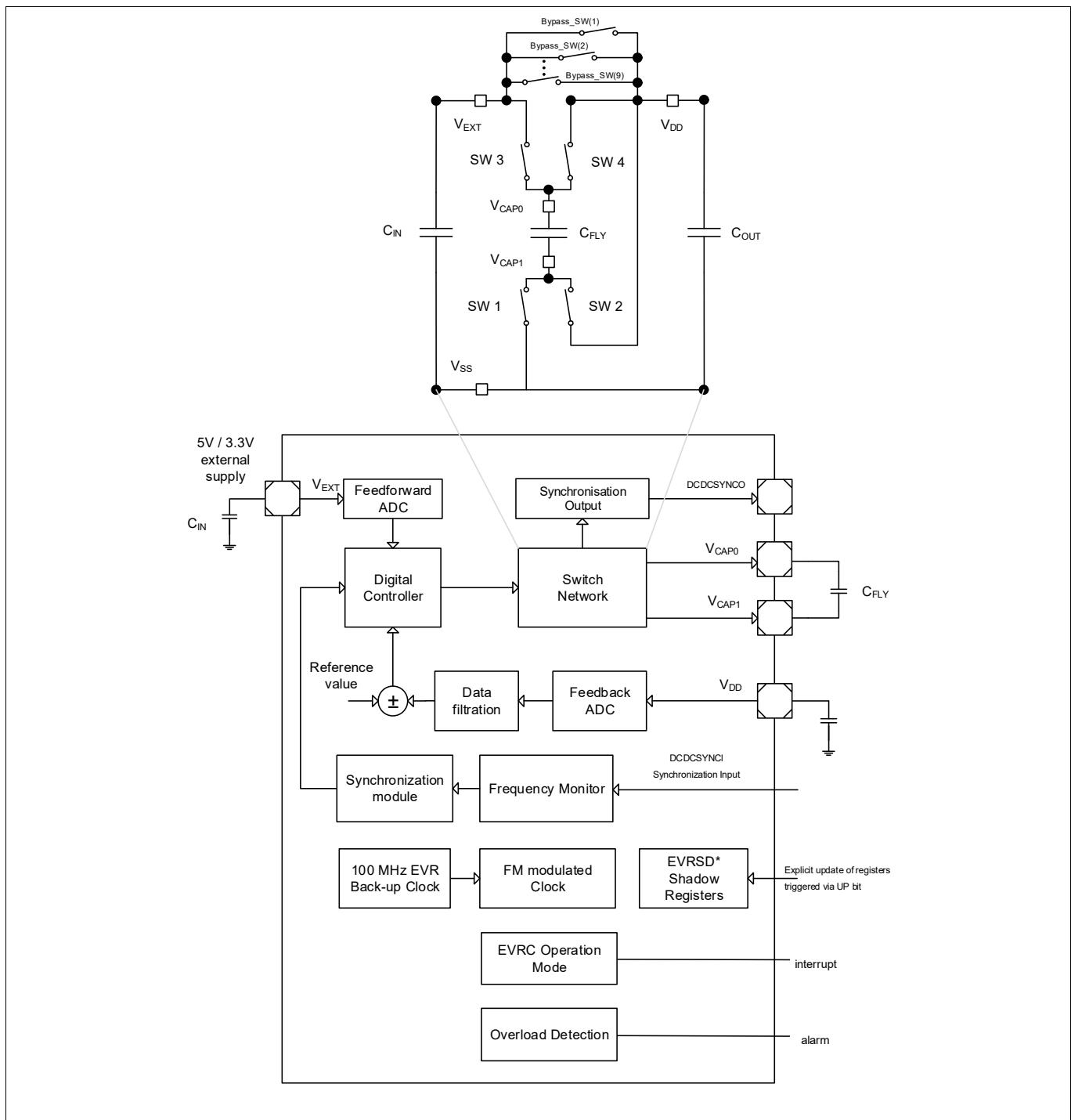


**Figure 129 EVR33 LDO regulator**

## Power Management System for Low-End (PMSLE)

### 12.2.2.2 Switch Capacitor Regulator (EVRC)

The Switch Capacitor SMPS regulator provides a higher efficiency of power conversion compared to the linear voltage regulator concept. However it requires additional external components and injects more switching noise into the system. The integrated switch capacitor regulator modulates internal switches to buffer the energy in capacitors in order to generate a regulated core supply. A flying capacitor and a buffer capacitor is required as shown in [Figure 130](#).



**Figure 130 EVRC Step down regulator**

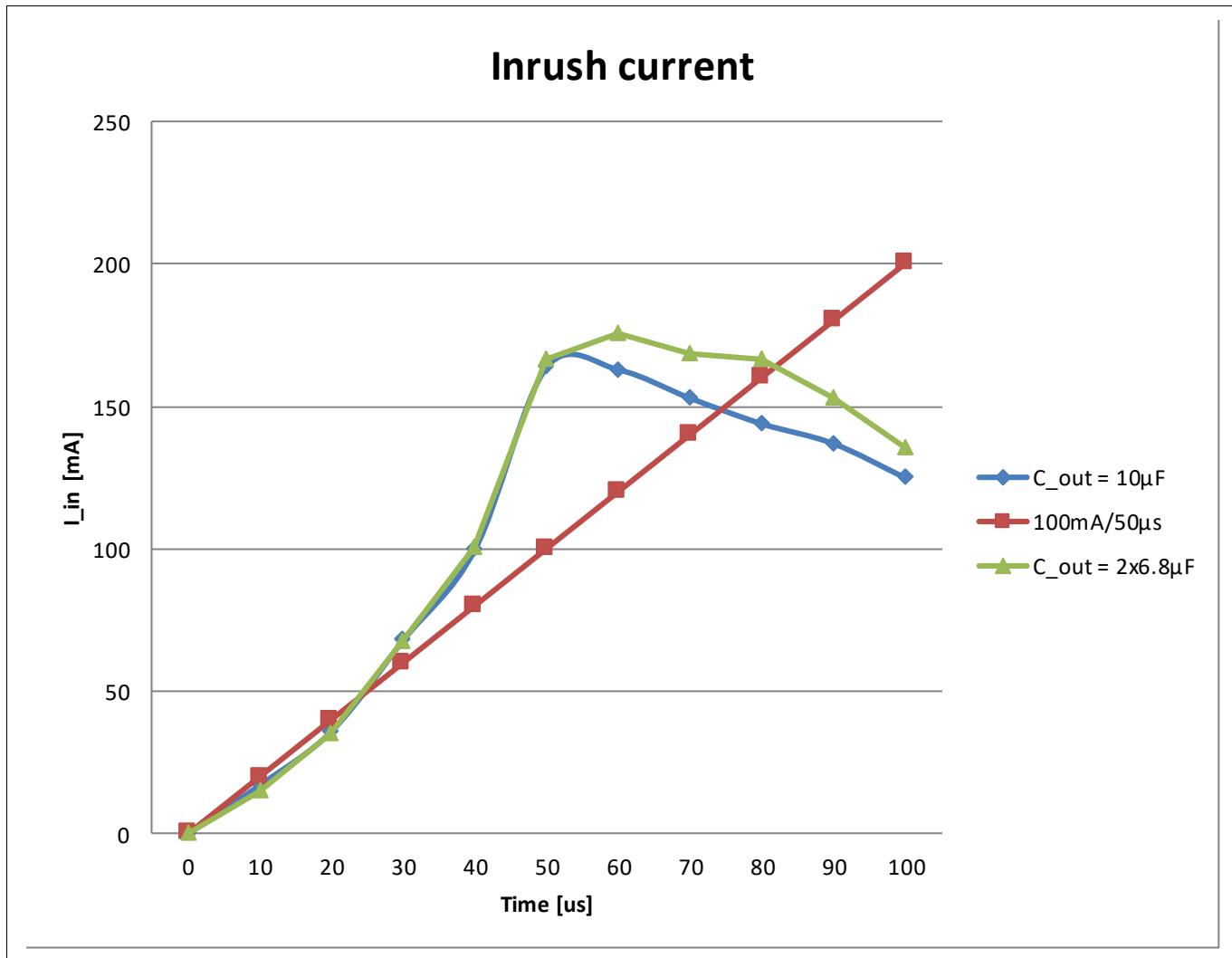
The control strategy involves synchronous switching of the 4 switches at a defined switching frequency in conductance modulation mode thus charging and discharging the switch capacitor network. The regulator

## Power Management System for Low-End (PMSLE)

operates with a fixed conversion ratio of 1/2 in SC mode. The recommended nominal switching frequency is 1.85 MHz in PWM mode and is derived based on efficiency, performance and EMI/EMC trade-off. The nominal switching frequency is configured with a 10 ns switching period resolution at a base frequency of 100 MHz back-up clock. The charge / discharge time are programmable via **EVRSDCTRL1.TON** and **TOFF** fields which inturn determine the switching frequency. The charge and discharge time are kept equal for best ripple performance. The output voltage is measured via the SMPS feedback ADC module and based on the deviation to the reference voltage the controller modulates the conductance value for the next charge / discharge period. The measured output voltage is then fed into the digital filter and provided to the digital controller. The measured core voltage is indicated in **EVRSDSTAT0.ADCFBCV** status bits. The target of the digital controller is to compute the new conductance control output based on the feedback. The conductance value of the previous period is indicated in **EVRSDSTAT0.CONDUCTANCE** status bits. The parameters for the digital controller are programmable. The external VEXT supply is also measured by the Primary SWD/VEXT Monitor ADC to facilitate a parameter switch incase the voltage crosses a threshold and to differentiate between 5 V or 3.3 V external supply case.

During the start-up phase, a soft-startup control strategy is used in order to avoid an overshoot of the output voltage and the charging of the buffer capacitor takes place gradually. During the initial switching periods a different set of coefficients are used in open loop operation. The conductance value is gradually changed to avoid current overshoots and keep the current jumps within 100 mA/50 us. The default switching frequency is 1.85 MHz during open loop start-up phase. The open loop operation ends when the ADC indicates that the output voltage is slightly below the target value. At this point the digital controller is re-configured and the normal closed loop operation begins. After a voltage transient (typically an overshoot), EVRC is ready and regulator output voltage is ok as indicated via **EVRSTAT.SDVOK**. The start-up phase and VCAPx behavior is portrayed in [Figure 128](#). The parameters of the step down regulator is consequently updated by the Firmware after reset release to achieve a more accurate EVRC output voltage and improved performance. The step-down regulator is also later programmed with the values enumerated in [Table 385](#) so as to match the application needs and the components used. A complete parameter update is triggered explicitly by writing to **EVRSDCTRL0.UP** bit and it need to be ensured that all the registers are consistent before triggering the update. The parameter update is not allowed during Start-up Mode. The droop compensation request and droop level value are taken immediately without waiting for a parameter update. During consequent supply ramp-down phase, the step down regulator ensure a graceful shut-down devoid of output voltage overshoots beyond absolute max ratings.

## Power Management System for Low-End (PMSLE)



**Figure 131 EVRC Inrush Current with different  $C_{out}$  values**

The conductance of the switches is, in a first order approximation, proportional to the current provided to the load. The switches are implemented as MOS transistors and the resistance of the active switches sets the maximum current capability. The output voltage VDD is measured with a 8 bit SMPS tracking ADC synchronous to the PWM and sampled twice every switching period. The digitized output voltage is then fed into the digital filter and the new conductance value is appropriately calculated by the PI controller. ADC samples are measured at the end of the charging and discharging phases corresponding to the double sampling scheme. The parameters for the PI controller are programmable.

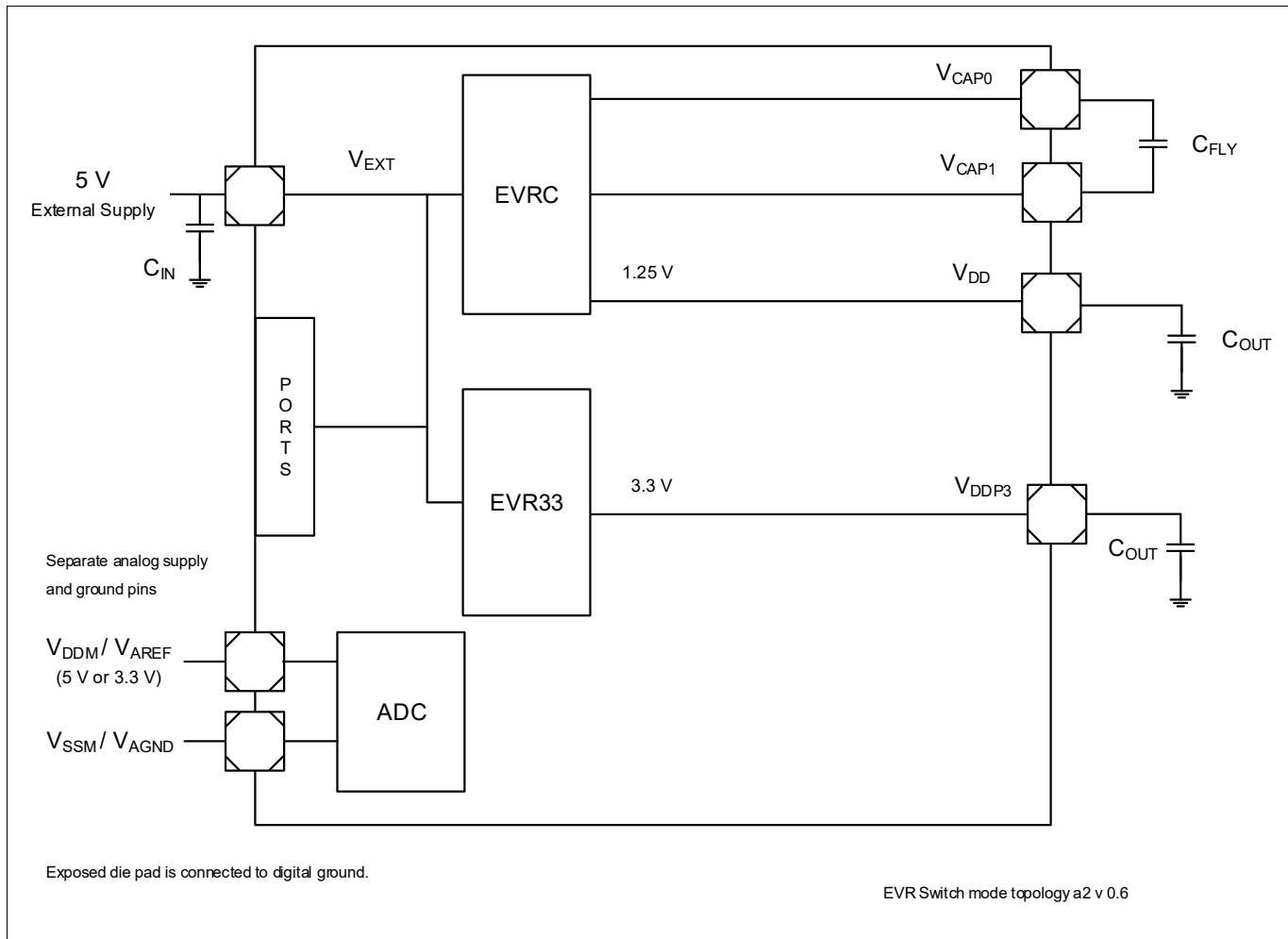
Maximum load current capabilities are limited in SC operational mode by the on-resistance of the switches (implemented using MOS transistors). In order to supply higher load currents, an additional set of bypass switches are implemented, as shown in Figure 11. A total of 9 bypass switches are implemented, adding increasingly higher load current capabilities as follows:

- bypass switch 1, adding 11 mA load current capability
- bypass switch 2, adding 22 mA load current capability
- bypass switches 3-9, each of them adding 44 mA load current capability

The step-down regulator may be informed on anticipated load jumps so that adequate preparation can be made. The controller could lower or raise the output voltage to compensate and thus minimise voltage over-/undershoots owing to a sudden load jump. The management of voltage droop is described in Power Management [Section 12.2.3.5](#)

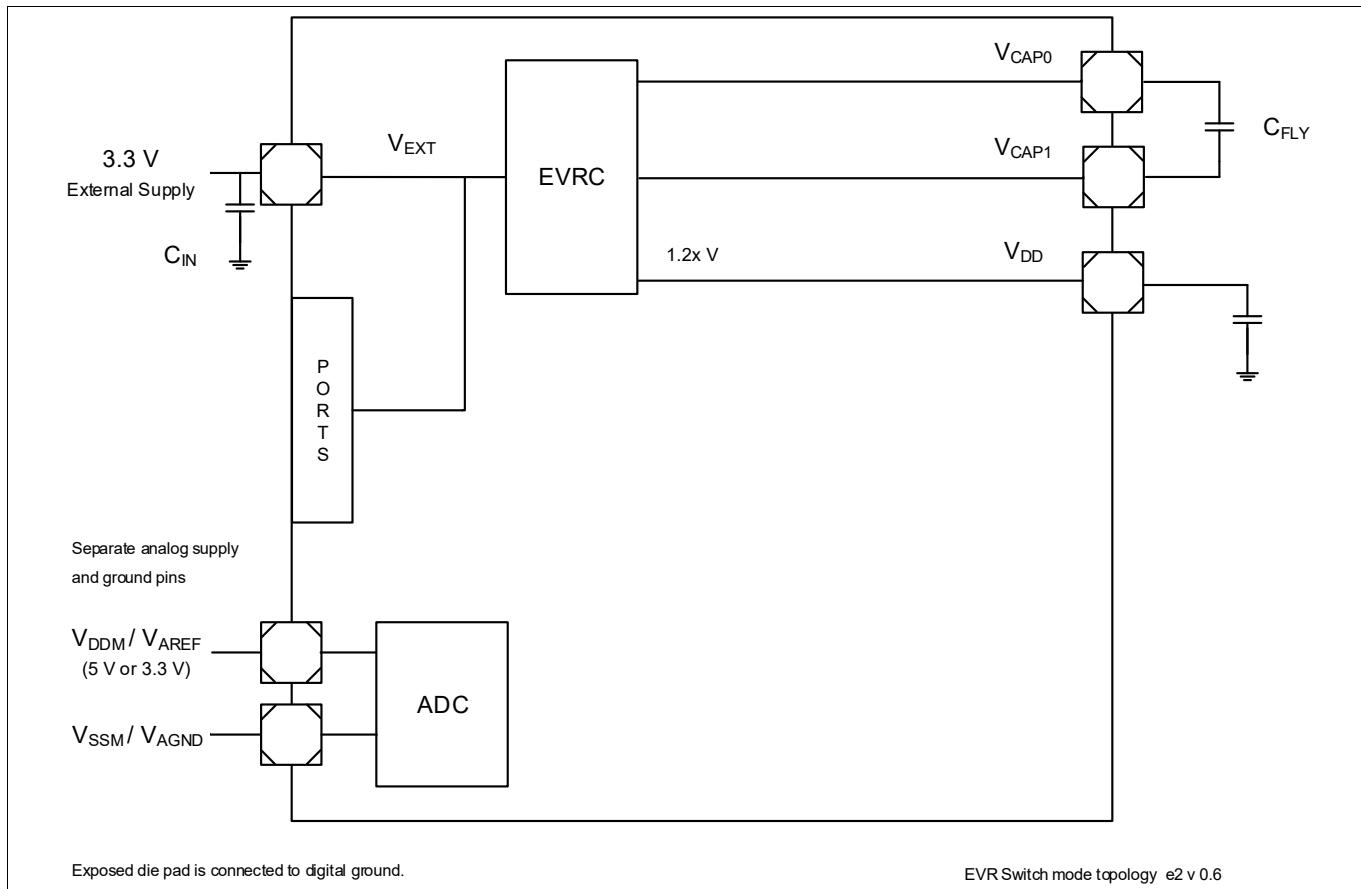
## Power Management System for Low-End (PMSLE)

In case frequency spreading is activated, the charge and discharge times are appended with a uniform random offset once per switching period. This allows to randomize the switching frequency of the SC DCDC regulator within a maximum value. The maximum random offset is bounded by the programmed value in **EVRSDCTRL0.SDFREQSPRD** register field. It is possible to synchronize an external step-down regulator which may run at much lower switching frequencies than the internal regulator. A scaled synchronisation clock output is provided and routed to an external pin (DCDCSYNC output).



**Figure 132 EVR Switch mode topology (a) - 5 V single supply**

## Power Management System for Low-End (PMSLE)

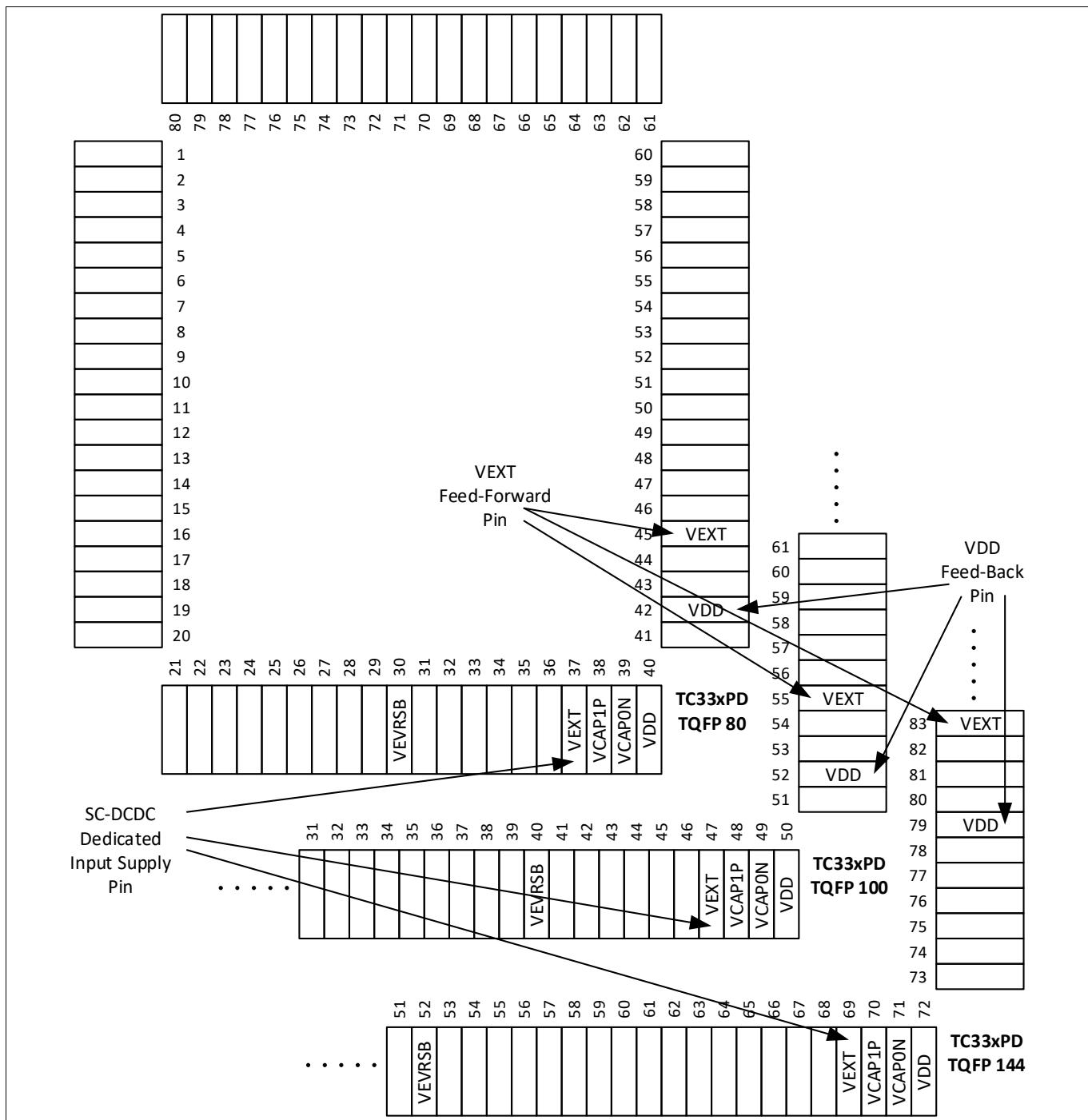


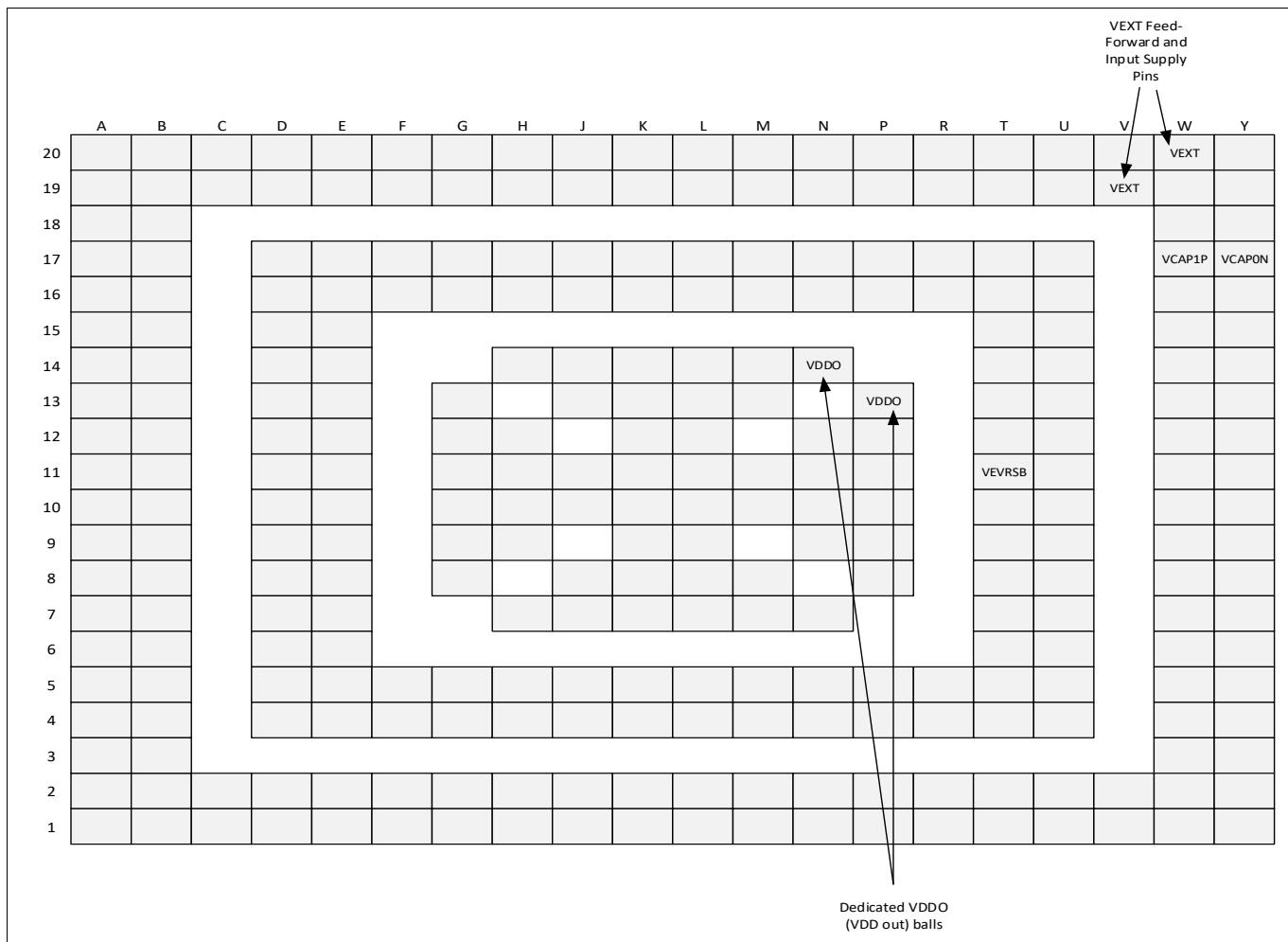
**Figure 133 EVR Switch mode topology (e) - 3.3 V single supply**

In case the current consumption in SC SMPS mode exceeds the switching regulator capabilities as indicated in the datasheet (IMAXSC parameter), bypass switches are automatically activated and a consequent interrupt (SRC\_PMS1) is issued to the system. The switch from the normal SC mode (EVRCMOD = 00b) to bypass switches enabled (EVRCMOD = 01b) is also indicated via **EVRSTAT.EVRCMOD** status bits. The lower bit of **EVRSTAT.EVRCMOD[12]** signal is used to generate the interrupt on SRC\_PMS1 interrupt node on a transition. Toggling of the **EVRSTAT.EVRCMOD** lower bit is expected for load currents around the IMAXSC threshold and multiple interrupts can be triggered as a result.

### 12.2.2.2.1 EVRC Supply Pins

The EVRC SC-DCDC pins, for the QFP and BGA packages, are shown in [Figure 134](#) and [Figure 135](#).

**Power Management System for Low-End (PMSLE)**

**Figure 134 EVRC SC-DCDC Supply Pins for QFP Packages**

**Power Management System for Low-End (PMSLE)**

**Figure 135 EVRC SC-DCDC Supply Pins for BGA Package (LFBGA292)**

In BGA packages, there is no dedicated VDD feedback ball. Instead, the VDD sense pin is connected inside the package to the common VDD plane in the redistribution layer (RDL).

#### 12.2.2.2.2 VDD Connectivity

It shall be ensured that all the VDD pins are connected together on the PCB. The VDD output pin is separated from the VDD feedback sense point pin, as shown in [Figure 123](#).

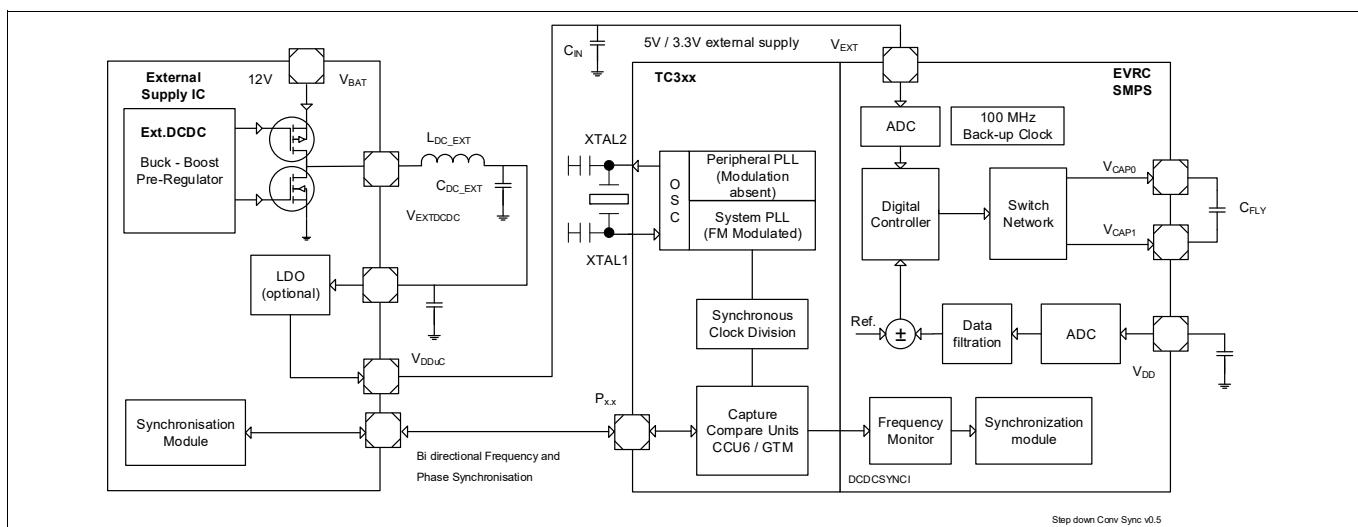
## Power Management System for Low-End (PMSLE)

### 12.2.2.2.3 EVRC Frequency and Phase Synchronization to CCU6/GTM Input

A synchronization input (DCDCSYNCI) is provided to the EVRC SMPS regulator from CCU6 or GTM module to synchronize the frequency and the phase of the internal EVRC regulator to the external DC DC regulator. The CCU6 / GTM module provides / captures two phase synchronised PWM signals; one PWM output to the internal EVRC regulator and the other as either input from or output to an external DC DC regulator using a Port pin as shown in **Figure 136**. The synchronization should only be enabled in trimmed mode, not during start-up or untrimmed mode. The pre-conditions for synchronization are:

- **EVRSDCTRL1.SYNCEN = 1<sub>B</sub>** - synchronization module enabled
- EVRC has finished the start-up phase and operates in trimmed mode.
- The synchronization input frequency range supported is between 1.6 MHz and 2 MHz. The nominal frequency of the synchronization input is 1.85 MHz. Bi-directional signal to/from external regulator is provided by the CCU6 or GTM unit.
- For correct edge detection, the duty cycle of the input synchronization signal shall be at least 20 ns (i.e. two 100 MHz clock cycles).

To disable the synchronization mode, SW shall configure **EVRSDCTRL1.SYNCEN = 0<sub>B</sub>**. The synchronization input source can be selected either from CCU6 or GTM inputs via the **EVRSDCTRL2.SYNCMUXSEL** bit-field.



**Figure 136 EVRC Synchronization Input**

The frequency monitor allows a maximum tolerable deviation of the synchronization input as configured in **EVRSDCTRL2.SYNCMAXDEV** register bit fields. A hysteresis is applied for the locking and unlocking, such that toggling lock behavior is avoided around the frequency monitoring limits as configured in **EVRSDCTRL2.SYNCHYST** hysteresis width register bit field. The DCDC locks to the synchronization signal if its period is within the following range:

- Sync. signal period  $\leq$  540 ns + **EVRSDCTRL2.SYNCMAXDEV** \* 10 ns - **EVRSDCTRL2.SYNCHYST** \* 10 ns
- Sync. signal period  $\geq$  540 ns - **EVRSDCTRL2.SYNCMAXDEV** \* 10 ns + **EVRSDCTRL2.SYNCHYST** \* 10 ns

Synchronization is unlocked when the synchronization signal leaves the hysteresis range, i.e.:

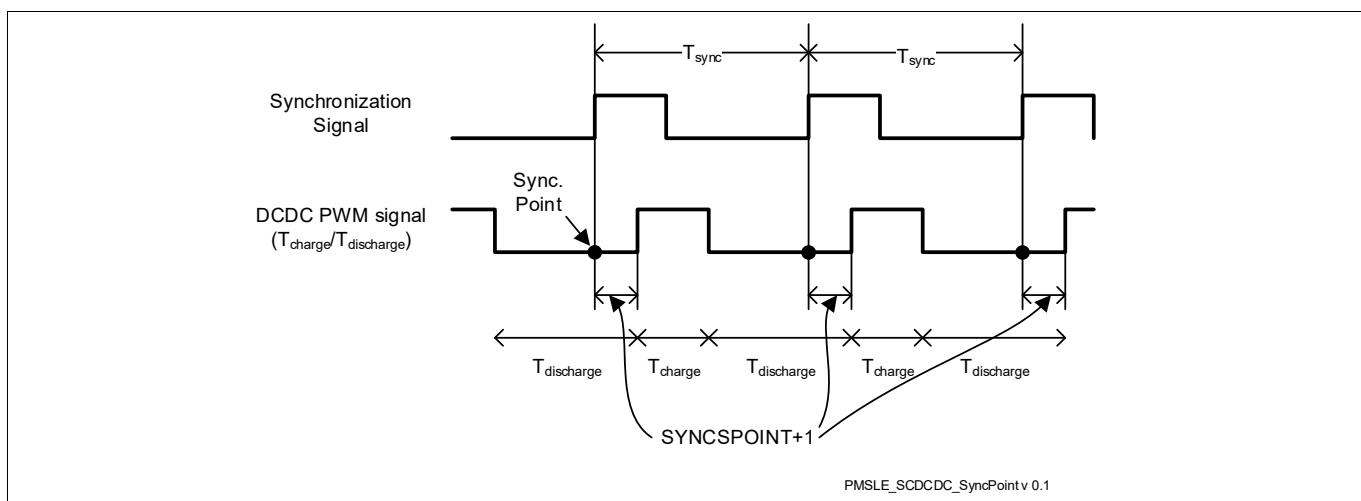
- Sync. signal period  $\geq$  540 ns + **EVRSDCTRL2.SYNCMAXDEV** \* 10 ns
- Sync. signal period  $\leq$  540 ns - **EVRSDCTRL2.SYNCMAXDEV** \* 10 ns

Here, the 540 ns represent the period of the nominal 1.85 MHz frequency (i.e. 54 cycles of 100 MHz) and the SYNCMAXDEV and SYNCHYST parameters are expressed in cycles of 100 MHz (i.e. number of 10 ns slots). The maximum deviation **EVRSDCTRL2.SYNCMAXDEV** can be specified up to 31 and the hysteresis range **EVRSDCTRL2.SYNCHYST** can be specified up to 7.

## Power Management System for Low-End (PMSLE)

The status of the synchronization lock is indicated via **EVRSTAT.SYNCLCK** bit. The loss of Synchronization Lock event is indicated by an interrupt which can be enabled via the **PMSIEN.SYNCLCK** register bit.

A programmable delay is introduced between the rising edge of the external synchronization signal and the rising edge of the DCDC PWM signal, required to safely turn off the discharge switches and perform calculations for the next switching cycle (FB-ADC sampling, conductance evaluation, bypass switch update etc.). This delay can be programmed by the SW in the **EVRSDCTRL6.SYNCSPOINT** bit-field and is expressed as a number of 100 MHz cycles (number of 10 ns slots). The rising edge of the synchronization signal will then be synchronized to the rising edge of the DCDC PWM signal - **EVRSDCTRL6.SYNCSPOINT** - 1, thus the delay between the signals is **SYNCSPOINT+1** cycles. This delay defines the Synchronization Point and is illustrated in [Figure 137](#).



**Figure 137 Synchronization Point between External Synchronization Signal and DCDC PWM Signal**

The synchronization point shall be set during the discharge phase, to allow turning off the discharge switches. The minimum setting for **EVRSDCTRL6.SYNCSPOINT** is 7, thus the minimum delay between the signals is 8 cycles (i.e. 80 ns). The reset value of **EVRSDCTRL6.SYNCSPOINT** is 8 and the SW shall never program a value below 7. The maximum delay is determined by the programmed discharge time (see the **EVRSDCTRL1.TOFF** setting).

### Synchronization Lock Procedure

- After the completion of the start-up phase, all EVRC parameters are configured as per the intended configuration. The **EVRSDCTRL2.SYNCMAXDEV** and **SYNCHYS** bit fields are configured. Frequency spreading, if required, can be activated in the **EVRSDCTRL0.SDFREQSPRD** register field. If frequency spreading is enabled (i.e. **EVRSDCTRL0.SDFREQSPRD > 0**), it is applied also during the synchronized state, however by increasing or decreasing only the charge phase by a random number of 100 MHz cycles within the programmed range (since the rising edges of the PWM and the input signal must remain in sync).
- A synchronization input signal is provided and configured in GTM / CCU6 module and is selected via **EVRSDCTRL2.SYNCMUXSEL** bit field. GTM LCDCDCOUT signal is selected via the LCDCDCOUTSEL register in GTM module. In case of CCU60, only COUT63 is routed. A phase shifted signal may be provided to the external DCDC regulator.
- When no load / line transients are ongoing, **EVRSDCTRL1.SYNCEN** bit is set to initiate the Synchronization Lock procedure.
- The frequency of the incoming synchronization signal is monitored for a single period consequently after **EVRSDCTRL1.SYNCEN** bit is set. At least one input period is required to evaluate whether the input frequency is valid. A parameter update is required to transfer the synchronization enable information.
- There are two locking options available, configurable via the **EVRSDCTRL6.SYNCLCKOPT** bit:
  - **EVRSDCTRL6.SYNCLCKOPT = 0<sub>B</sub>** - fast locking option (typical duration is 1-2 switching cycles)

## Power Management System for Low-End (PMSLE)

- **EVRSDCTRL6.SYNCLCKOPT = 1<sub>B</sub>** - slow locking option (typical duration is 4-8 switching cycles)
- Fast locking can cause an undershoot of the output voltage. The reset setting is 1<sub>B</sub> (slow locking) and is the recommended option for all applications where the synchronization locking time is not critical, to avoid the possible undershoots. A smooth synchronization is achieved by optimizing the frequency and phase correction depending on the moment of the rising edge of the synchronization signal.
- When the synchronization is completed, the DCDC switching frequency and phase is altered and locked to the incoming signal.
- When the DCDC switching frequency is locked to the synchronization input, the **EVRSTAT.SYNCLCK** status bit is set into locked state indicating the completion of the lock procedure.

### Synchronization Un-Lock Procedure

- The **EVRSDCTRL1.SYNCEN** bit shall be reset to initiate the Synchronization unlock procedure.
- As soon as synchronization is disabled via **EVRSDCTRL1.SYNCEN**, the **EVRSTAT.SYNCLCK** status bit toggles into unlocked state. The Synchronization unlock interrupt is generated if enabled (via **PMSIEN.SYNCLCK**).
- The synchronization input signal is consequently deactivated from GTM / CCU6 module via **EVRSDCTRL2.SYNCMUXSEL** bit field. The phase shifted signal to the external DCDC regulator from GTM / CCU6 is also deactivated.

### Synchronization Un-Lock Event

- In case the frequency of the input synchronization signal violates the allowed limits (respectively more than nominal period + maximum deviation or is less than nominal period - maximum deviation), the EVRC unlocks and continues operation with the programmed charge/discharge times.
- As soon as the synchronization is lost, the **EVRSTAT.SYNCLCK** bit toggles into unlocked state. The Synchronization unlock interrupt is generated if enabled (via **PMSIEN.SYNCLCK**).
- A re-lock is triggered when the incoming signal period is less than (nominal period + maximum deviation - sync hysteresis) or more than (nominal period - maximum deviation + sync hysteresis).

## Power Management System for Low-End (PMSLE)

### 12.2.2.3 Components and Layout

The efficiency of EVR is influenced by the characteristics of the selected components and also the placement and routing of the components on the PCB. The additional external components for the SC SMPS regulator constitute a flying capacitor (1 uF) and a buffer capacitor (10 uF). An input capacitor (4,7 uF) is required in case of SC SMPS mode to limit the input current ripple. The trace impedances and distances to the external components should be in principle as small as possible. The component requirements are documented in the datasheet and recommendations are also provided in application notes.

In case of usage of Emulation devices, it should be taken care that the component choice also considers the current additionally drawn by Emulation RAM and additional modules.

Component characteristics would be recommended in the datasheet and is also documented in [Table 385](#).

It shall be ensured that all the VDD pins are connected together when using SC DCDC regulator. The VDD output pin adjacent to the VCAP pins is separated from the VDD feedback sense pin as shown [Figure 123](#) and needed to be connected together for proper functioning of regulator closed loop.

It should be taken care that each supply pin in QFP packages or a pair of supply pins in BGA packages has a decoupling capacitor close to the pins. Supply pins belonging to a common supply rail shall be connected together after the respective decoupling capacitors and shall be buffered by an additional larger capacitor based on the constraints of the regulator which supplies the rail. In case of EVR33 and EVRC regulator, recommended buffer capacitors are enumerated in [Table 385](#). It should be taken care to have a low trace resistance to the decoupling capacitors and buffer capacitors for better performance and EMI / EMC behavior. The dimensioning of the buffer capacitors is based predominantly on the load jumps triggered during reset events and stability criteria of the regulator.

**Table 385 EVRC Regulator Component Reference**

No.	Condition	Optimal Register Value (Modes a & e)	Components (Package)
1.)	IDD < 400 mA fDCDC = 1,85 MHz VEXT < 3.3 V	Use default reset values.	Output Capacitor - (10 uF nominal) CGA6M3X7R1C106K (1210) - t.b.c. Input Capacitor - (4.7 uF nominal) CGA4J1X7R1E475K (0805) - t.b.c. CGA4J3X7R1C475K (0805) - t.b.c. Flying Capacitor - (1 uF nominal) CGA3E1X7R1E105K (0603) - t.b.c. CGA3E1X7R1C105K (0603) - t.b.c.
2.)	IDD < 400 mA fDCDC = 1,85 MHz VEXT < 5 V	Use default reset values.	Output Capacitor - (10 uF nominal) CGA6M3X7R1C106K (1210) - t.b.c. Input Capacitor - (4.7 uF nominal) CGA4J1X7R1E475K (0805) - t.b.c. CGA4J3X7R1C475K (0805) - t.b.c. Flying Capacitor - (1 uF nominal) CGA3E1X7R1E105K (0603) - t.b.c. CGA3E1X7R1C105K (0603) - t.b.c.

**Table 386 EVR33 External Component Reference**

No.	Condition	Optimal Register Values	Components (Package)
1.)	IDDP3 < 100 mA		Output Buffer capacitor (1 uF) - C3216X7R1C105K

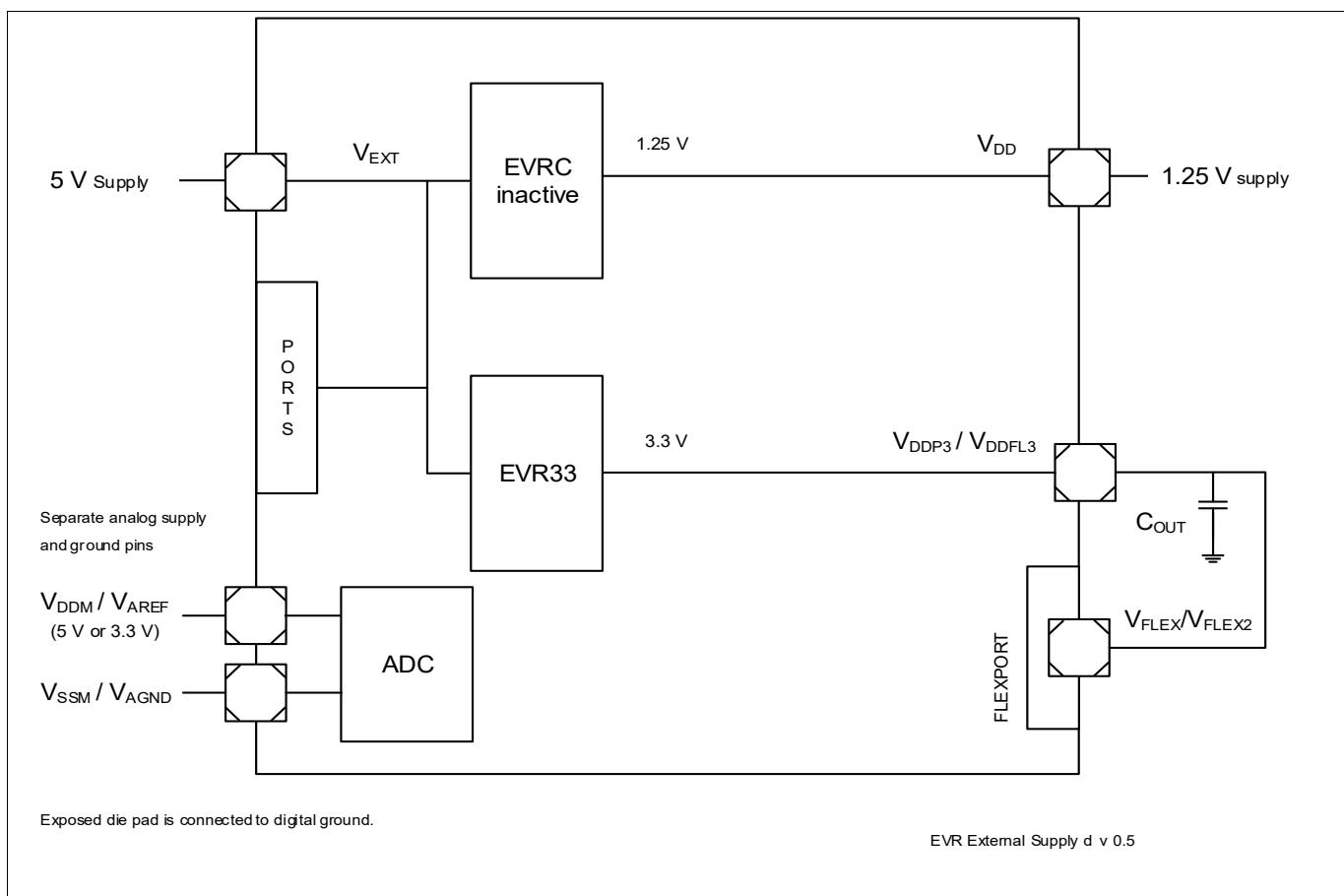
## Power Management System for Low-End (PMSLE)

### 12.2.2.4 External Supply Modes

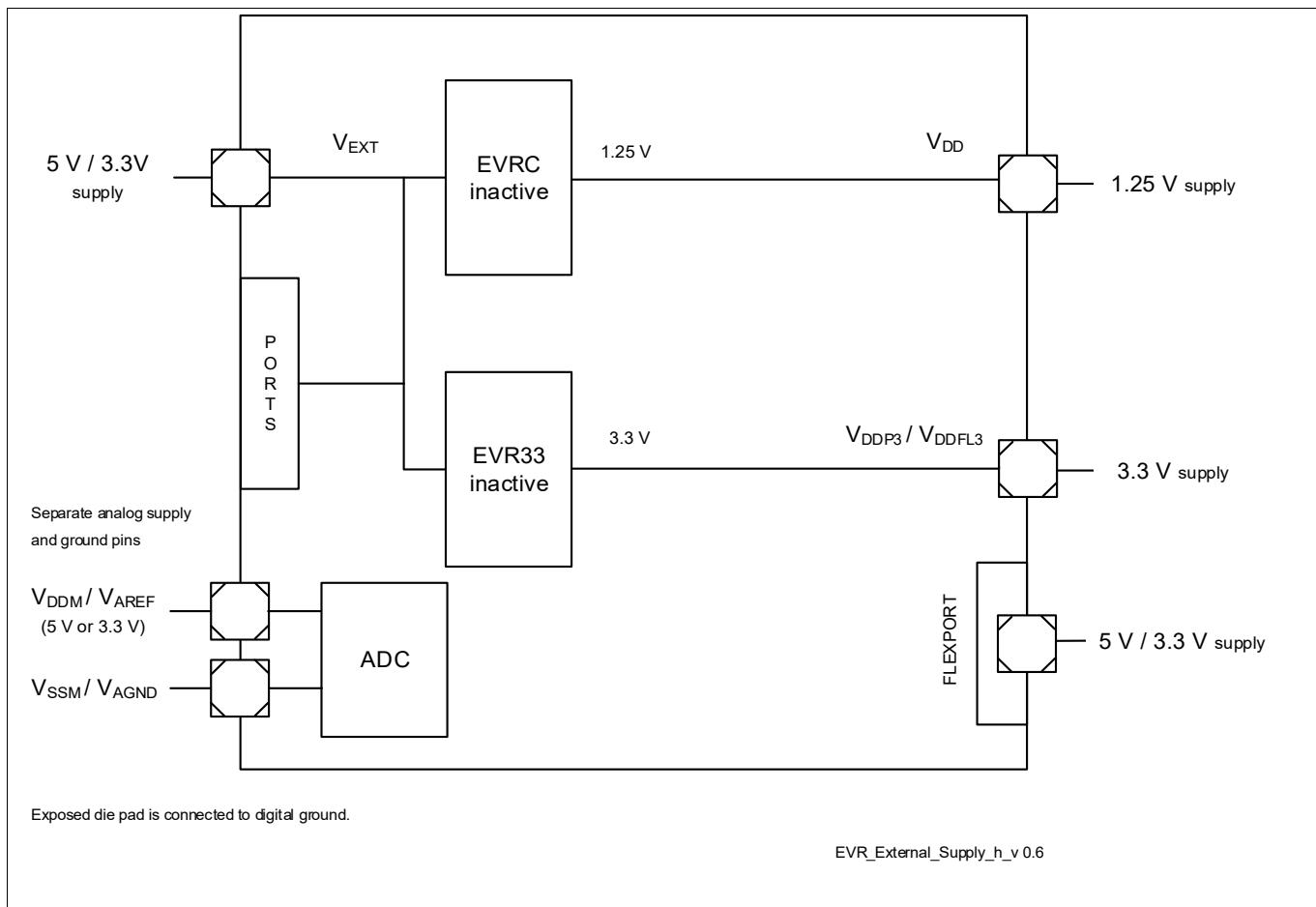
The external supply modes involve deactivating any or both of the EVRC and EVR33 regulators. In this mode, EVR33 is disabled via the HWCFG[1] configuration pin and the EVRC is disabled via the HWCFG[2] configuration pin respectively.

Following external supply modes are supported.

- VEXT = 5 V and VDD supplied externally. VDDP3 is generated using the EVR33 regulator as shown in [Figure 138](#).
- VEXT = 5 V or 3.3 V and VDDP3 is supplied externally. VDD is generated using the EVRC regulator.
- VEXT, VDDP3 and VDD are all supplied externally as shown in [Figure 139](#).



**Figure 138 External Supply mode (d) - VEXT and VDD externally supplied**

**Power Management System for Low-End (PMSLE)**

**Figure 139 External Supply mode (h) -  $V_{EXT}$ ,  $V_{DDP3}$  and  $V_{DD}$  externally supplied**

## Power Management System for Low-End (PMSLE)

### 12.2.2.5 Supply Voltage Monitoring

The PMS module implements a staggered voltage monitoring build upon a primary and a secondary monitor providing adequate redundancy to meet safety requirements. The primary monitor ensures that the micro controller is put into a cold PORST reset state when the lowest operational voltage thresholds are violated. The secondary monitor serves as an additional safety monitor providing over- and under-voltage alarms for multiple supply rails. Monitors are realized using dedicated 8 bit ADC converters and result comparators.

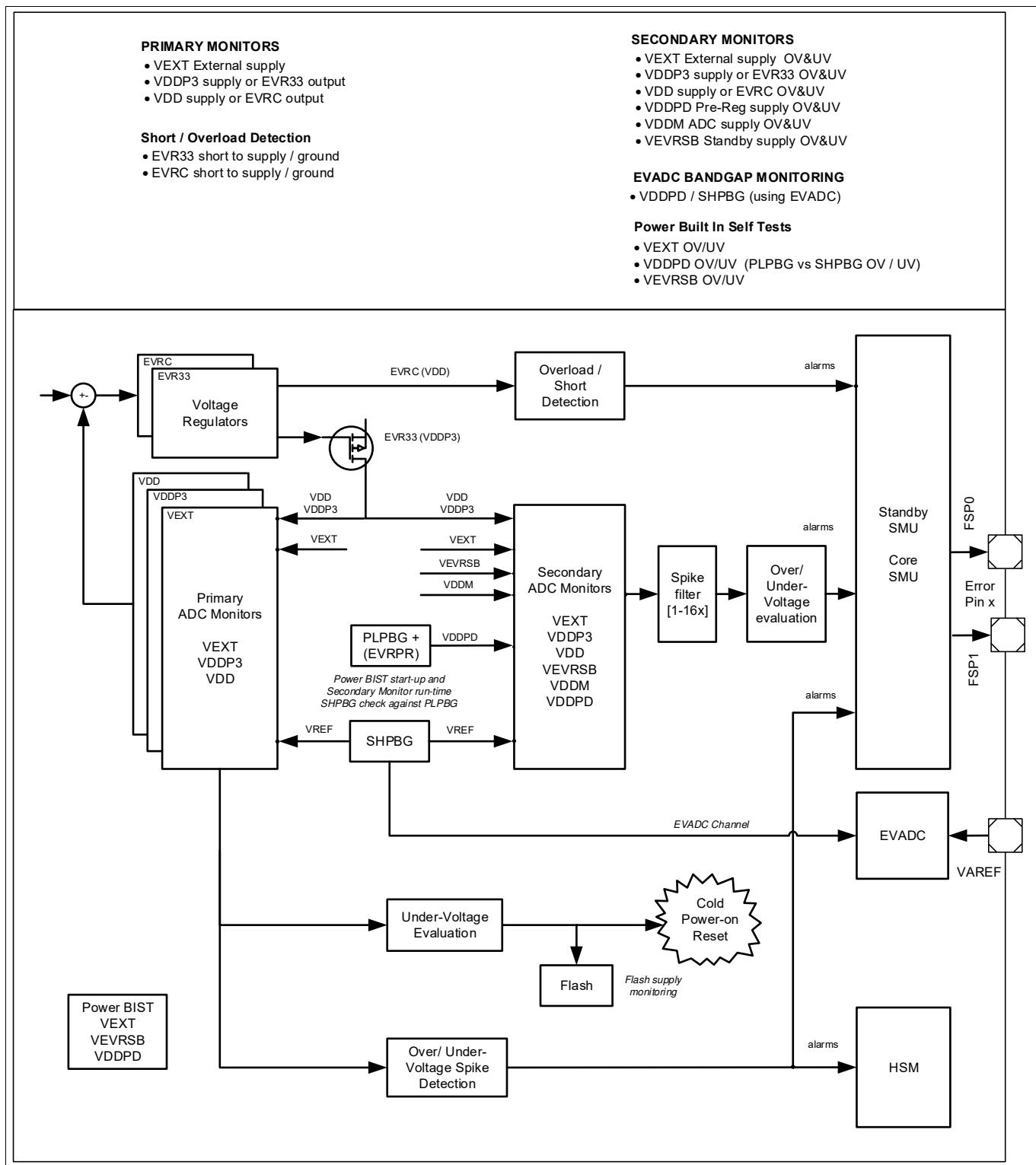


Figure 140 Supply Monitor Overview

## Power Management System for Low-End (PMSLE)

### 12.2.2.5.1 Primary under-voltage monitors and Cold PORST

Primary under-voltage monitoring of the external VEXT supply, VDDP3 / EVR33 supply and VDD / EVRC supply are inherently carried out to ensure proper functioning of the system. The thresholds for the primary monitors represent the lowest possible thresholds for the correct functioning of the system. The threshold and tolerance is documented in datasheet as VxxPRIUV parameter. In case of violation of these thresholds, cold PORST is activated and PORST pin is pulled low (strong current sink) thus setting the device into reset state. Following the reset release and firmware boot, it can be inferred from STBYR, EVRC, EVR33 or SWD bits in RSTSTAT register as to whether the violation of these thresholds led to the previous reset. The primary under-voltage monitoring is kept active even if the respective EVRs have been disabled and the supply is provided externally as shown in **Table 387**. The thresholds are trimmed and the monitoring is activated or deactivated via the **EVRRSTCON** register. The user shall not modify the default values of the **EVRRSTCON** register, as any alteration of the primary reset monitoring violates the operational conditions of the microcontroller and may lead to unexpected behavior during the dynamic undershoot regulation or during the power down sequence.

The cold PORST is asserted when the supply voltage drops below **EVRRSTCON.RSTxTRIM** value. During cold PORST reset release, to avoid consecutive toggling PORST during slow supply ramp-ups, a voltage hysteresis is supported. The cold PORST is de-asserted or released when the supply rises above (**EVRRSTCON.RSTxTRIM** + Hysteresis) value. The PORST pin is driven low for a minimum nominal time of 10 us on recognition of cold PORST irrespective whether the voltages have been immediately restored so that there is adequate time to recognise it externally.

Further more, additional power-on detectors are available for VEVRSB supply (supplied by VEXT), VEXT supply (supplied by VEVRSB) and VDDPD internal supply (via VDDPD POR monitor) to ensure a proper minimum-power detection, robust start-up and standby operation. Undervoltage of VDDPD internal supply and external VEXT supply will lead to the assertion of the LVD (Low Voltage Detector) reset. Undervoltage of external VEVRSB supply will lead to the assertion of the LVD reset indirectly via the VDDPD POR monitor as VDDPD is generated from VEVRSB. Assertion of LVD reset is reflected in RSTSTAT.STBYR bit and can be evaluated in the next start-up. After a normal supply start-up, only STBYR and PORST bits in RSTSTAT register are set.

The primary Supply WatchDog (SWD monitor) monitors the ramp-up of external VEXT supply voltage and keeps the micro controller in cold Power On Reset state as long as the supply has not reached the operational region. Likewise, it also allows detecting ramp-down or brown out conditions of external supply so that the device can be brought into a cold Power On Reset state when the voltage has dropped below the lowest operational threshold. Nevertheless, It is recommended to monitor externally all supplies generated external to the micro controller and to assert PORST reset pin in case of violation of the lowest operational limits. The pass device dropout voltage should be taken into consideration when setting these limits. In case of 5 V nominal external supply and 3.3 V in turn being generated by the internal EVR33 LDO regulator, the external supply shall maximum drop during normal RUN mode considering adequate pass device dropout as documented in datasheet.

The external VEXT supply, VDD / EVRC and VDDP3 / EVR33 supplies are measured by Primary Monitor ADCs and the measured value is updated in **EVRADCSTAT** register after conversion completion at every PMS clock cycle.

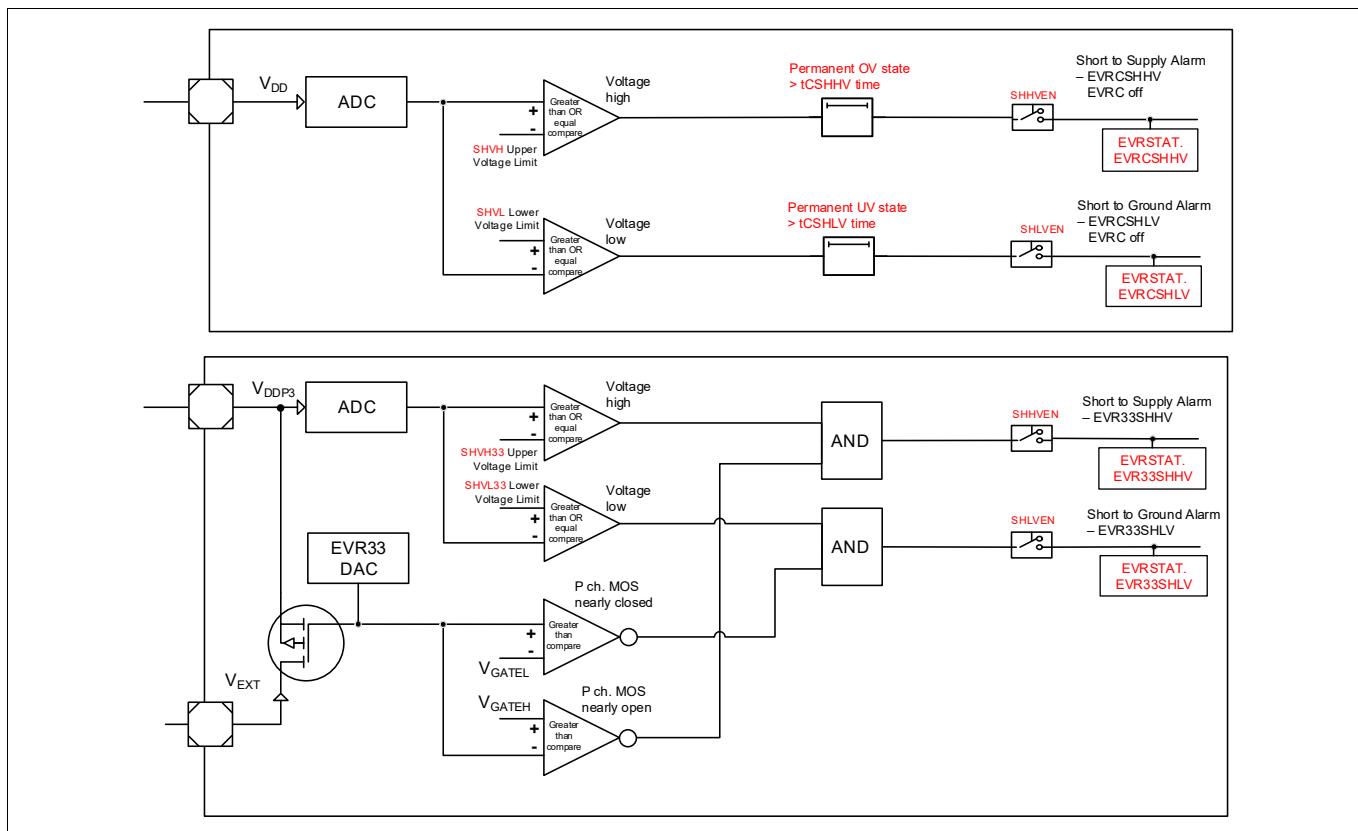
In case of primary monitor violation, respective status bits are set to indicate the event as shown in **Table 387**. These bits maybe evaluated during consequent start-up after cold power-fail reset to recognize which among the supply rails had the power-fail.

The violation of the primary under-voltage and over-voltage operational limits of VEXT, VDDP3 and VDD supply rails is communicated to the HSM module. HSM module may lock access to EVR registers via SLCK bit so that supply generation cannot be influenced by other masters. This is to ensure that trojan programs do not manipulate the supplies to gain access to the system. VEXT, VDDP3 and VDD rail primary monitor measurements are compared with **HSMOVMON** and **HSMUVMON** thresholds and alarms are routed to HSM module and to the SMU (as shown in **Figure 140**). The violations are indicated in **EVRADCSTAT** status flags. The unfiltered primary monitor ADC measurements are used to detect power spikes on the main supply rails and consequently alarms are provided to HSM and SMU. Each primary monitor ADC tracking speed is bounded by the maximum supply

## Power Management System for Low-End (PMSLE)

slope of a single LSB step every nominal 25 MHz ADC clock cycle. This results in a maximum tracking speed of 500V/ms (20mV LSB/40ns) for VEXT SWD primary monitor, 375V/ms (15mV LSB/40ns) for VDDP3 primary monitor. A voltage based short detection scheme is enabled for EVRC via **EVRSDCTRL9.SHLVEN** / **SHHVEN** register bit fields. The short detection scheme for EVRC output is as portrayed in [Figure 141](#). VDD FBADC result is compared against SHVL and SHHV thresholds. If the low voltage or high voltage condition occurs continuously for more than tCSHLV or tCSHHV duration, the respective voltage alarms are activated and are indicated by **EVRSTAT.EVRxSHHV** and **EVRSTAT.EVRxSHLV** register status bits. If the low voltage or high voltage condition disappears before tCSHLV or tCSHHV expiry, then tCSHLV or tCSHHV timers are reset. The recovery from EVRC short switch-off state is possible only with a renewed ramp-up of VEVRSB and VEXT supply rails. EVRC Short signal is filtered for 6 consecutive values using a spike filter and the filtered signal leads to EVRC switch off.

A short detection scheme may be activated for EVR33 via **EVR33CON.SHLVEN** / **SHHVEN** bits. The short detection scheme for EVR33 is portrayed in [Figure 141](#). Short to higher voltage is deduced when the voltage regulator control output or pass device gate voltage has saturated at the lower limit and at the same time the regulator voltage output has crossed the absolute maximum limit. Short to lower voltage is deduced when the voltage regulator control output or pass device gate voltage has saturated at the upper limit and at the same time the regulator voltage output has stayed at the minimum limit. In both cases the respective alarms are activated and indicated by **EVRSTAT.EVR33SHHV** and **EVRSTAT.EVR33SHLV** register bits.



**Figure 141** Short to Supply and Ground Detection

## Power Management System for Low-End (PMSLE)

### 12.2.2.5.2 Secondary over- and under-voltage monitors and alarm generation

Additional secondary over-voltage and under-voltage monitoring against programmable thresholds is provided for all supplied and generated voltages. The secondary monitors are based on a secondary bandgap reference independent from the primary band-gap reference. The monitored voltages include the external VEXT supply voltage, VDDP3 / EVR33 supply, VDD / EVRC supply, external VEVRSB supply voltage, external VDDM ADC supply voltage and the internally generated VDDPD Pre- Regulator output voltage as shown in [Figure 143](#) and [Figure 144](#). The secondary voltage monitors are kept active even if the respective EVRs have been disabled and the supply is provided externally as shown in [Table 387](#). In case of a threshold violation, an SMU alarm event is generated. The threshold and tolerance is documented in datasheet as VxxMON parameter.

The secondary monitor violation is notified depending on the direction of voltage transition as programmed in [EVROMONCTRL](#) register. The appropriate thresholds for voltage monitoring can be programmed in the [EVROVMON](#), [EVROVMON2](#), [EVRUVMON](#) and [EVRUVMON2](#) registers. These can be calculated by linear interpolation based on multiple voltage levels and corresponding thresholds provided in VxxMON datasheet parameters. In case of an active monitoring violation, respective status flags are set in the [EVRSTAT](#) register. It can be inferred from OVC, OV33, OVSWD, OVPRE, OVS and OVDDM bits in [EVRSTAT](#) register as to whether over-voltage thresholds for the respective voltage domains were violated. Likewise, it can be inferred from UVC, UV33, UVSWD, UVPRE, UVSB and UVDDM bits in [EVRSTAT](#) register as to whether under-voltage thresholds were violated. The respective status bits may be evaluated to differentiate between an over-voltage or an under-voltage event and to recognize which supply rail had triggered the alarm event to SMU as shown in [Table 387](#). The secondary monitor measurement latency to measure all 6 supply rails is documented in datasheet as tMON parameter. The supply rails are converted one after another in a continuous scan mode. It is also possible to deactivate individually the secondary monitors in [EVROMONCTRL](#) register. If the respective OVMOD and UVMOD bits are set to 00, then the ADC conversion for the particular supply rail is skipped by the Secondary Monitor and (tMON/6) time is respectively reduced from the total conversion time.

The [EVRUVMON2.VDDMLVLSEL](#) bit-field shall not be modified by the application software (as it is not related to the secondary monitoring thresholds). The application SW shall always read out the default value of [EVRUVMON2.VDDMLVLSEL](#) and write it back unmodified together with any new undervoltage monitoring threshold information in the [EVRUVMON2](#) register.

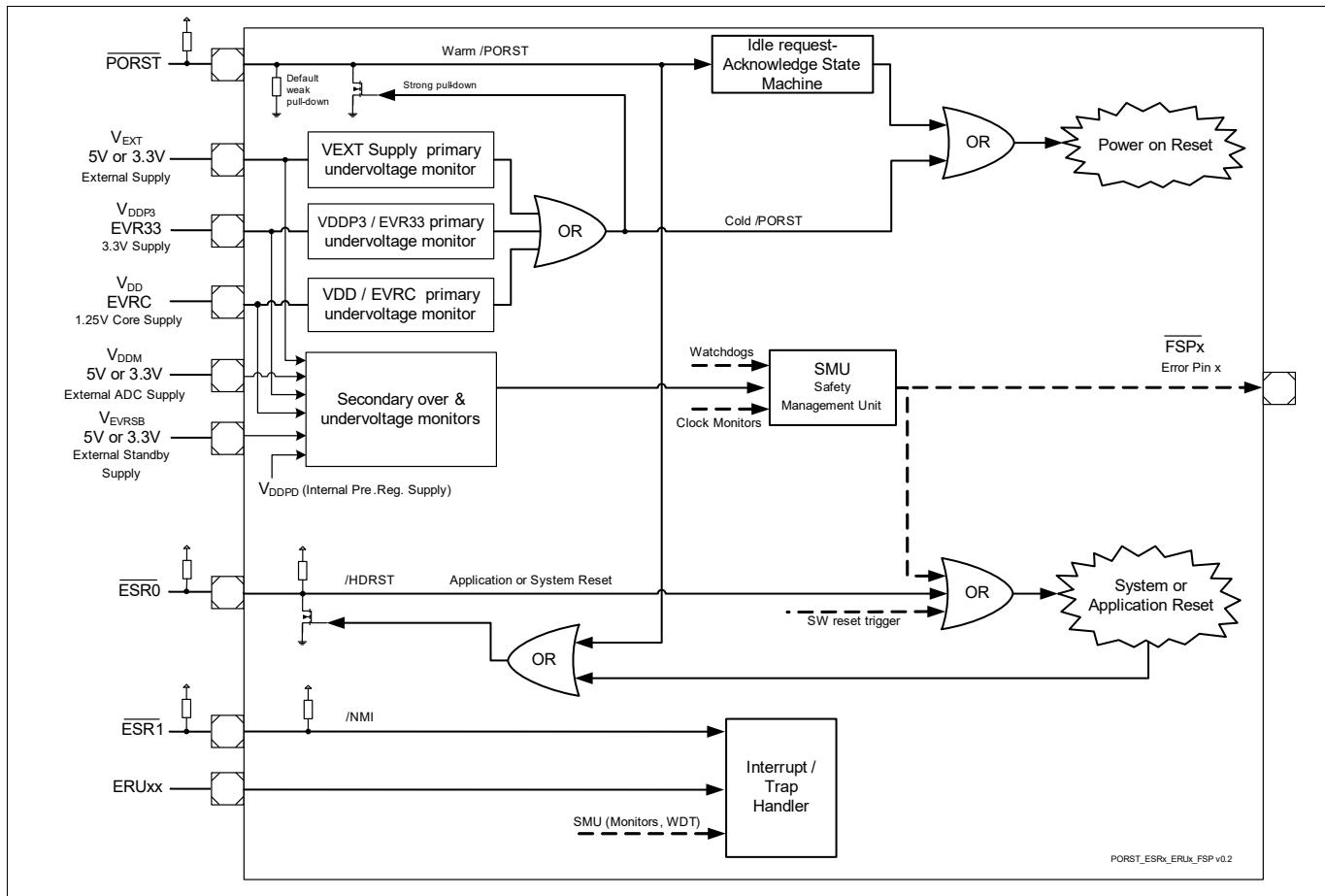
The monitored voltages, namely the VEXT, VDDP3, VDD, VDDPD, VEVRSB, and VDDM supplies are measured by Secondary Monitor ADCs and the actual measured value is updated in [EVROMONSTAT1](#) and [EVROMONSTAT2](#) register after conversion completion at regular intervals. Spike filtering of consecutive ADC results are used to generate alarm to SMU and also used for the filtered values indicated in [EVROMONSTAT1 / EVROMONSTAT2](#) registers as configured via adjustable filter coefficients in [EVROMONFILT.xx FIL](#) bit fields. In case VDDM supply voltage drops below 500 mV outside the operational limits, then Secondary monitor stops converting and the activity counter [EVROMONSTAT1.ACTVCNT](#) freezes at the last value.

In case of over-voltage supply alarms, it may be ensured that the supply to the device is switched off to avoid damage. The Error Pin Fail Safe Protocol ensures that the over-voltage condition is communicated to the external regulator even when TC3xx is in warm reset state.

After start-up, it may happen that supply over- or under-voltage alarms may already have been triggered depending on residual start-up voltages or supply dynamics. Likewise during [EVROMONCTRL](#) or [EVROMONFILT](#) reconfigurations, spurious alarms may be raised depending on filter state and changed configuration. Therefore before activating SMU alarm generation or triggering latent fault supply alarm tests, the secondary monitors and filters need to be completely reset. It need to be ensured that SMU alarms and associated interrupts are foremost deactivated in [EVROMONCTRL](#) / [EVROMONFILT](#) / [PMSIEN](#) registers, then filters are cleared via [EVROMONFILT.CLRFIL](#) = 1, alarms and interrupts are then consequently re-configured to the intended voltage level and filter settings in [EVROMONCTRL](#) / [EVROMONFILT](#) registers followed by activation of filters via [EVROMONFILT.CLRFIL](#) = 0. A delay time of 4 us has to be awaited before alarm activation after configuration is changed in [EVROMONCTRL](#) / [EVROMONFILT](#) registers.

## Power Management System for Low-End (PMSLE)

In case of application and system resets, PMS alarms happening during the respective reset shutdown and release will be reflected in SMU\_stdby AGX alarm status registers and consequently SMU\_stdby FSP reaction may be triggered if so configured in SMU\_stdby AGFSP.FEx registers. On the contrary, PMS alarms occurring during warm reset phase will be not be latched in SMU\_core AGX alarm status registers as they are in reset state. Furthermore, alarms which have occurred during the reset phase would not be consequently forwarded to the SMU\_core on reset release.



**Figure 142 Monitoring and Reset Pins**

**Power Management System for Low-End (PMSLE)**
**Table 387 Voltage Monitoring**

<b>Supply Pin / Rail</b>	<b>Primary Under-voltage Monitor State (ON/OFF) Status Registers set on Under-voltage</b>	<b>Secondary Over &amp; Under-voltage Monitor State (ON/OFF) Status Registers</b>	<b>Supply Range V</b>	<b>Is the Pin supplied</b>
RUN or SLEEP system mode during supply modes a,d,e & h.				
V <sub>EXT</sub>	<p>ON. RSTSTAT.SWD set if V<sub>EXT</sub> drops below VEXTPRIUV limit triggering cold PORST. During cold start-up on an initial V<sub>EXT</sub> ramp-up, RSTSTAT.SWD is not set.</p> <p>RSTSTAT.PORST bit implicitly set as cold PORST would trigger also warm PORST.</p> <p>RSTSTAT.STBYR set if V<sub>EXT</sub> drops below VLDRST5 voltage limit.</p> <p>EVRSTAT.RSTSVD shows current status. EVRADCSTAT.ADCSVD shows ADC result. EVRADCSTAT.OVSWD (HSM &amp; SMU alarm) EVRADCSTAT.UVSWD (HSM &amp; SMU alarm)</p>	<p>ON EVRSTAT.OVSWD (SMU alarm) EVRSTAT.UVSWD (SMU alarm) EVROMONSTAT1.ADCSVD</p>	2.97-5.50 V	External 5V or 3.3V Supply to be provided
V <sub>DDP3</sub>	<p>ON RSTSTAT.EVR33 set if V<sub>DDP3</sub> drops below VDDPRIUV limit triggering cold PORST. During cold start-up on an initial V<sub>DDP3</sub> ramp-up, RSTSTAT.EVR33 is not set.</p> <p>RSTSTAT.PORST bit implicitly set as cold PORST would trigger also warm PORST.</p> <p>EVRSTAT.RST33 shows current status. EVRADCSTAT.ADC33V shows ADC result. EVRADCSTAT.OV33 (HSM &amp; SMU alarm) EVRADCSTAT.UV33 (HSM &amp; SMU alarm)</p>	<p>ON EVRSTAT.OV33 (SMU alarm) EVRSTAT.UV33 (SMU alarm) EVROMONSTAT1.ADC33V</p>	2.97-3.63 V	EVR33 active or external 3.3V supply to be provided
V <sub>DD</sub>	<p>ON RSTSTAT.EVRC set if V<sub>DD</sub> drops below VDDPRIUV limit triggering cold PORST. During cold start-up on an initial V<sub>DD</sub> ramp-up, RSTSTAT.EVRC is not set.</p> <p>RSTSTAT.PORST bit implicitly set as cold PORST would trigger also warm PORST.</p> <p>EVRSTAT.RSTC shows current status.</p> <p>EVRADCSTAT.ADCCV shows ADC result. EVRADCSTAT.OVC (HSM &amp; SMU alarm) EVRADCSTAT.UVC (HSM &amp; SMU alarm) (RSTSTAT.PORST bit implicitly set)</p>	<p>ON EVRSTAT.OVC (SMU alarm) EVRSTAT.UVC (SMU alarm) EVROMONSTAT1.ADCCV</p>	1.125-1.375 V	EVRC active or external 1.25V supply to be provided

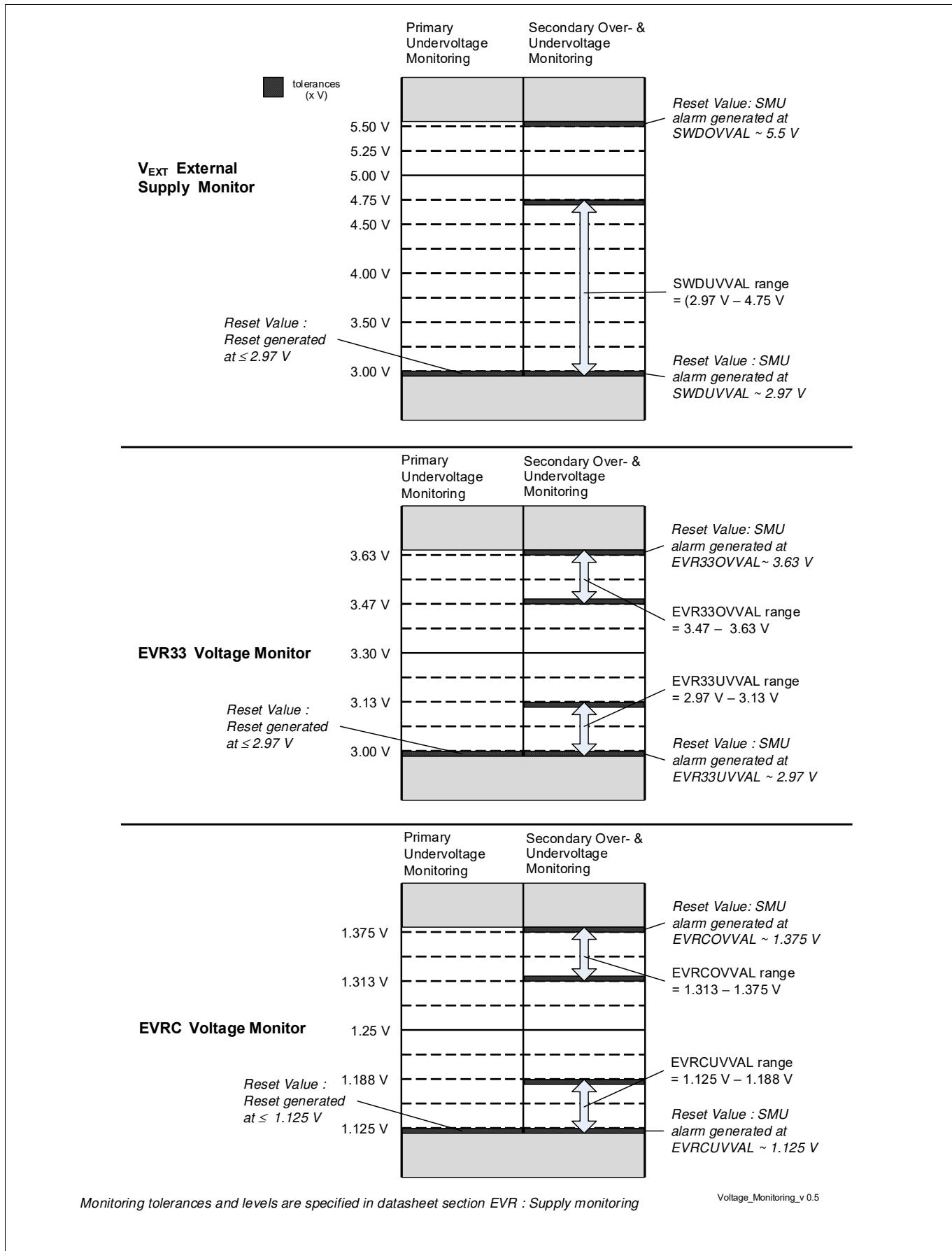
**Power Management System for Low-End (PMSLE)**
**Table 387 Voltage Monitoring (cont'd)**

<b>Supply Pin / Rail</b>	<b>Primary Under-voltage Monitor State (ON/OFF) Status Registers set on Under-voltage</b>	<b>Secondary Over &amp; Under-voltage Monitor State (ON/OFF) Status Registers</b>	<b>Supply Range V</b>	<b>Is the Pin supplied</b>
$V_{EVRSB}$	ON.(via VEVRSB detector) RSTSTAT.STBYR set if $V_{EVRSB}$ drops below VLDRSTSB voltage limit triggering LVD reset. RSTSTAT.PORST bit implicitly set as LVD reset would trigger also warm PORST.	ON EVRSTAT.OVSB (SMU alarm) EVRSTAT.UVSB (SMU alarm) EVRMONSTAT2.ADCSB	2.97-5.50 V	External 5V or 3.3V EVR / Standby Supply to be provided
$V_{DDM}$	not available.	ON EVRSTAT.OVDDM (SMU alarm) EVRSTAT.UVDDM (SMU alarm) EVRMONSTAT2.ADCVDDM	2.97-5.50 V	External Supply to be provided
$V_{DDPD}$	ON.(via VDDPD POR detector) RSTSTAT.STBYR set if $V_{DDPD}$ drops below lowest voltage limit triggering LVD reset. RSTSTAT.PORST bit implicitly set as LVD reset would trigger also warm PORST.	ON EVRSTAT.OVPRE (SMU alarm) EVRSTAT.UVPRE (SMU alarm) EVRMONSTAT2.ADCPRE	1.125-1.375 V	Internal voltage not available on pin.

**STANDBY system mode during supply modes a,d,e & h.**

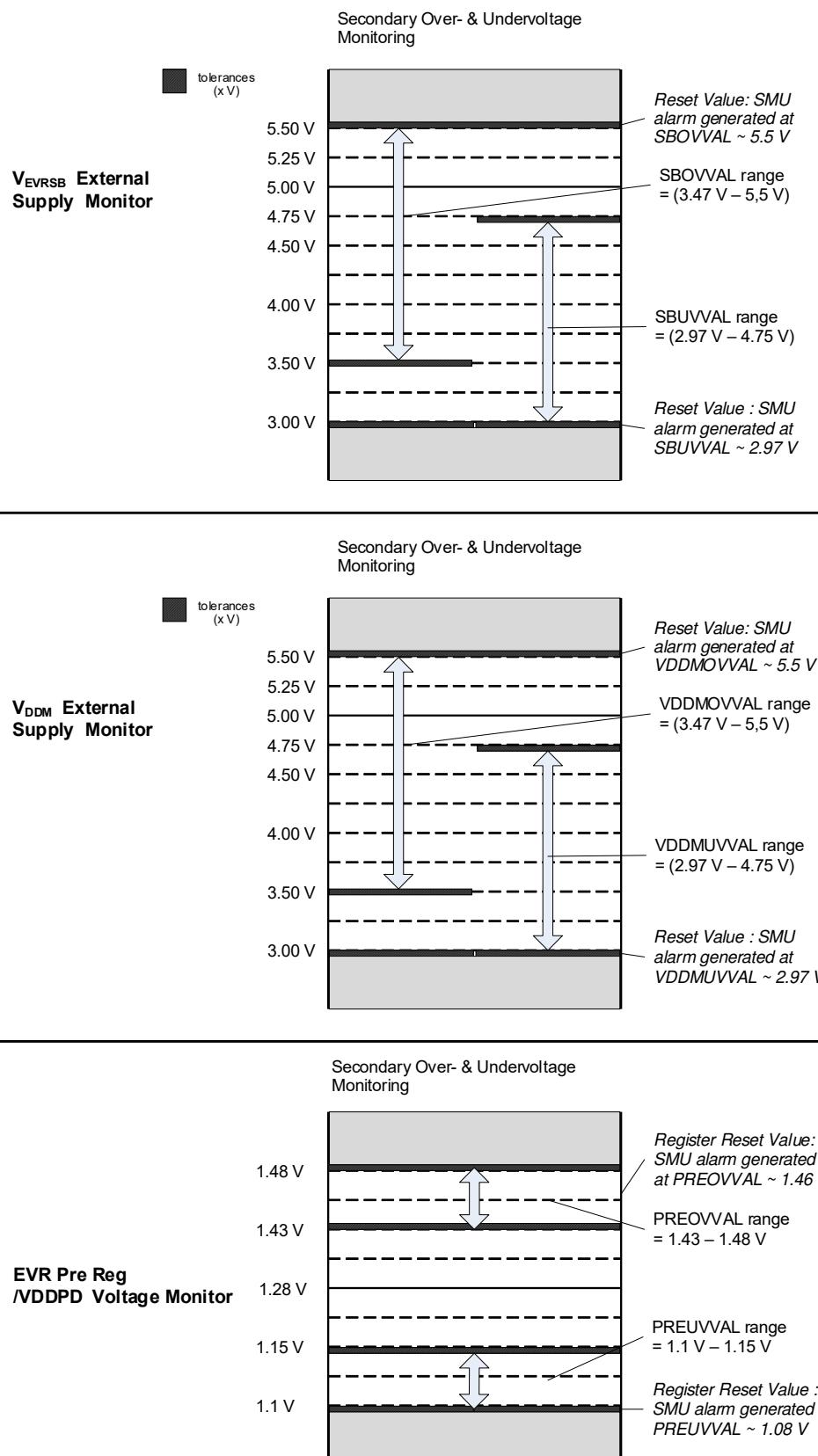
$V_{EXT}$	OFF/ON based on VEXTSTBYEN. RSTSTAT.STBYR set if $V_{EXT}$ drops below VLDRST5 voltage limit triggering LVD reset during Standby mode if VEXTSTBYEN = 0 & PWRWKEN = 0 is configured before Standby entry.  If Standby entry is triggered by power fail events; RSTSTAT.SWD, EVRC, EVR33 and RSTSTAT.PORST may be additionally set.	OFF	2.97-5.50 V	ON OFF if separate $V_{EVRSB}$ Standby supply used.
$V_{DDP3}$	OFF	OFF	0 V	OFF
$V_{DD}$	OFF	OFF	0 V	OFF
$V_{EVRSB}$	ON.(via VEVRSB detector) RSTSTAT.STBYR set if $V_{EVRSB}$ drops below VLDRSTSB voltage limit triggering LVD reset during Standby mode.	OFF	2.97-5.50 V	External Standby Supply to be provided
$V_{DDM}$	not available.	OFF	0-5.50 V	ON or OFF
$V_{DDPD}$	ON.(via VDDPD POR monitor) RSTSTAT.STBYR set if $V_{DDPD}$ drops below lowest voltage limit triggering LVD reset.	OFF	1.125-1.375 V	Internal voltage not available on pin.

## Power Management System for Low-End (PMSLE)



**Figure 143 Voltage Monitoring - VEXT, VDDP3 & VDD**

## Power Management System for Low-End (PMSLE)



Monitoring tolerances and levels are specified in datasheet section EVR : Supply monitoring

**Figure 144 Voltage Monitoring - VEVRSB, VDDM & VDDPD**

## Power Management System for Low-End (PMSLE)

### 12.2.2.5.3 Power Built In Self Test at Start-up (PBIST)

A Power Built-In-Self-Test (PBIST) at start-up allows the testing of supply levels, power functions and voltage monitors before cold PORST reset release.

The internal EVRPR Pre-regulator VDDPD voltage based on the primary low power bandgap (PLPBG) is tested using secondary monitor ADC against the secondary bandgap (SHPBG) at supply ramp-up. This allows to monitor the bandgap voltages against each other during start-up and the device continue to remain in reset state till the test has passed. During runtime, bandgap monitoring is realised by VDDPD monitoring using secondary monitor ADC and alarm is raised to SMU in case of VDDPD over and under-voltage event.

VEVRSB and VEXT voltage levels are checked using secondary monitor ADC before starting the regulators in PBIST state. In case the voltages are not within the limits, the device reset state is not deasserted. Furthermore, the PBIST test is passed and reset state is deasserted when VDDM supply voltage is above 500 mV.

After EVRC and EVR33 regulators are ramped up, additional overvoltage and undervoltage checks are carried out for VEVRSB ( $5,84V / 2,75V \pm 5\%$ ), VEXT ( $5,84V / 2,75V \pm 5\%$ ), VDDP3 ( $3,81V / 2,0V \pm 5\%$ ), VDD ( $1,46V / 1,0V \pm 5\%$ ) and VDDPD ( $1,46V / 1,0V \pm 5\%$ ) rails before cold PORST reset release in PBIST2 state. The limits are the default reset values of **EVROVMON**, **EVROVMON2**, **EVRUVMON** and **EVRUVMON2** registers.

A PBIST\_Off pad has been added to disable only the PBIST check as an implementation risk mitigation measure. The default state after reset is that the pad is pulled low which activates PBIST check during Start-up and Standby transitions. If the pad level is high, then the PBIST check is always skipped during Start-up and Standby PBIST and PBIST2 states.

### 12.2.2.5.4 Secondary Monitor and Standby SMU Built in Self Test (MONBIST)

After reset release, MONBIST for the secondary monitors and alarm generation path may be carried out by user software. Secondary Monitor BIST ensures a higher latent fault coverage for the secondary monitors and the associated alarm and error pin fault logic routed to the Standby SMU. The MONBIST can be triggered during start-up via MONBISTCTRL.TSTEN register bit in Standby SMU module. During ongoing MONBIST, PMS SFF test shall not be triggered. MONBIST test takes less than 25 us execution time. The procedure is as follows :

- The Standby SMU shall be enabled via SMUEN register bitfield for MONBIST functionality.
- The MONBISTCTRL.TSTCLR bit shall be set foremost to clear all the flags and reset the test logic. This clears TSTEN, TSTRUN, TSTDONE, TSTOK, SMUERR and PMSERR bits.
- **EVRMONFILT** is set to 0x20000000 to clear the filter and to activate 1 x spike filter.
- **EVRMONCTRL** is set to 0xa5a5a5 to activate Over-voltage and Under-voltage alarms.
- The corresponding Over-voltage and Under-voltage interrupts are disabled by clearing **PMSIEN**.OVx/UVx register bit fields.
- FSP reaction on alarms are disabled by setting AGFSP.FEx to 0.
- CMD.FSP0EN and CMD.FSP1EN configuration bits are cleared to avoid spurious Error pin activation during MONBIST.
- CMD.ASCE is set to ensure that all pending alarms are cleared in AGx registers.
- **EVRMONFILT** is set back to 0x00000000 before enabling MONBIST to ensure alarm propagation.
- Consequently the MONBIST is enabled via MONBISTCTRL.TSTEN register bit.
- The MONBISTSTAT.TSTRUN register bit is set to indicate an ongoing test by MONBIST logic.
- Once the test is completed, MONBISTSTAT.TSTDONE bit is set and MONBISTSTAT.TSTRUN bit is cleared.
- The MONBISTSTAT.TSTOK bit indicates that the test was successfully completed.
- The MONBISTSTAT.SMUERR and MONBISTSTAT.PMSERR bits indicate that errors were detected during the MONBIST.

---

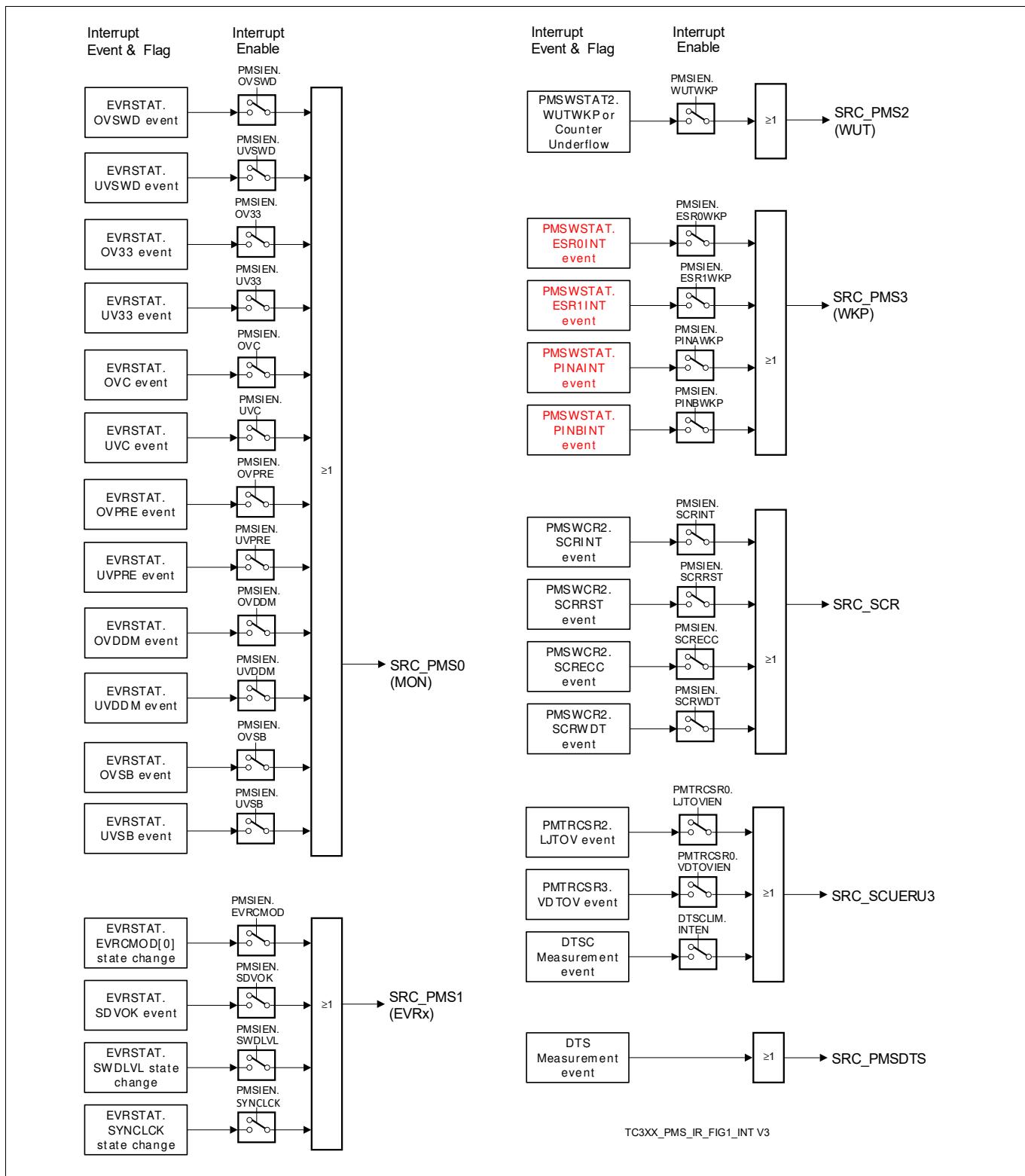
**Power Management System for Low-End (PMSLE)**

- FSPERR bit shall be cleared after MONBIST before enabling FSP reaction. If alarms happened during MONBIST, status registers may be updated and shall be cleared before Standby SMU initialization. TSTEN bit is cleared at the end of MONBIST.

## Power Management System for Low-End (PMSLE)

### 12.2.2.6 Interrupts

Following events may be configured to lead to interrupts routed to Interrupt Router in Normal Run and Sleep System modes. If enabled by the related interrupt enable bit in register **PMSIEN**, an interrupt pulse can be generated on one of the service request outputs (SRC\_PMS0, SRC\_PMS1, SRC\_PMS2, SRC\_PMS3, SRC\_SCR, SRC\_SCUERU3). Interrupts are forwarded from PMS to IR module within 4 fspb clock cycles after the occurrence of the event.



**Figure 145 Interrupt Sources and Events**

## Power Management System for Low-End (PMSLE)

### 12.2.2.7 OCDS Trigger Bus (OTGB) Interface

#### PMS OTGB Features

- Voltage signals and ADC outputs
  - Primary VDD, VDDP3 and VEXT voltage monitor outputs
  - Primary EVRC (SMPS) core voltage feedback ADC output
  - Secondary VDD, VDDP3 and VEXT voltage monitor outputs
  - EVRPR / VDDPD voltage monitor output
  - VEVRSB Standby supply voltage monitor output
  - VDDM ADC supply voltage monitor output
  - DTS temperature output
- EVR control outputs
  - EVR33 regulator DAC control output
  - EVRC switching control output driving the switches
  - EVRC Regulator output and internal signals
  - Wake-up timer count
  - PMS and SCR register interface signals

The PMS module has two 16 bit ([Table 388](#)) trigger sets which are selected with the **OTSS** register. The trigger sets can be arbitrarily mapped to OTGB0/1 busses. Refer OCDS chapter for more details.

**Table 388 PMS Trigger Sets**

Trigger Set	Details
<a href="#">TS16_ADCMON Monitor Trigger Set</a>	<a href="#">Table 389</a>
<a href="#">TS16_EVRCON Control Trigger Set</a>	<a href="#">Table 390</a>

The PMS trigger signals relate to the 100 MHz internal back-up clock, which can be different to the OTGB/OTGM clock. It should be taken care that the triggers and associated signals are synchronised to SPB clock domain.

#### 12.2.2.7.1 ADC Monitor and Voltage Trigger Sets

ADC Monitor Trigger Sets consist of the important voltage signals measured by various PMS ADC monitors. The multiplexer allows to map arbitrary and different signal groups to the high and the low byte of a 16 bit Trigger Set. In addition it is possible to use one or two 16 bit Trigger Sets with this flexibility. All this is controlled with **OTSCO**.

**Power Management System for Low-End (PMSLE)**
**Table 389 TS16\_ADCMON Monitor Trigger Set**

<b>Bits</b>	<b>Name</b>	<b>Description</b>
[7:0]	SG0	8 bit Analog output from selected Analog monitors
		PRADCCV Primary Core / VDD voltage monitor output
		PRADC33V Primary VDDP3 voltage monitor output
		PRADCSWDV Primary VEXT voltage monitor output
		PRADCFBCV Primary EVRC SMPS core voltage feedback output
		SECADCCV Secondary Core / VDD voltage monitor output
		SECADC33V Secondary VDDP3 voltage monitor output
		SECADCSWDV Secondary VEXT voltage monitor output
		SECADCPRE EVRPR / VDDPD voltage monitor output
		SECADCSB VEVRSB standby voltage monitor output
		SECADCVDDM VDDM ADC voltage monitor output
		DTSRESULTL DTS Temperature output [7:0]
		DTSRESULTH DTS Temperature output [11:8]
[15:8]	SG1	Independent selection with same options as for Bits [7:0]

**12.2.2.7.2 EVR Control output Trigger Sets**

EVRCON Control Trigger Sets consist of the important control outputs of various regulators in PMS subsystem. All this is controlled with **OTSC1**.

**Table 390 TS16\_EVRCON Control Trigger Set**

<b>Bits</b>	<b>Name</b>	<b>Description</b>
[15:0]	EVR33OUT	EVR33 regulator DAC control output
	CONDUCTANCE	EVRC conductance value as indicated in <b>EVRSSTATO.CONDUCTANCE</b>
	EVRCOUT	Array of EVRC regulator signals from the SMPS module selected via DMOND multiplexer.
	WUTCNT	Wake-up timer count ([23:15] reduced to 15th bit)
	TCINT [7:0] SCRINT [15:8]	PMS and SCR output and input bus interface

## Power Management System for Low-End (PMSLE)

### 12.2.3 Power Management

#### 12.2.3.1 Power Management Overview

The Power Management scheme allows activation of power down modes so that the system operates with the minimum required power for the corresponding application state. A progressive reduction in power consumption is achieved by invoking Idle, Sleep or Standby modes respectively. The Idle mode is specific to each individual CPU where as Sleep and Standby modes influence the complete system.

As shown in [Table 391](#), there are two power modes available for each CPU:

- CPU Run Mode
- CPU Idle Mode

**Table 391 CPU Power Management**

Mode	Description
<b>Run Mode</b>	The CPU clock is active and code is being executed.
<b>Idle Mode</b>	<p>CPU may enter Idle Mode on following events:</p> <ul style="list-style-type: none"> <li>• On a SW Idle request issued by setting register bits PMCSR<sub>x</sub>.REQSLP = 01<sub>B</sub> when CPU has no active tasks to perform.</li> <li>• On a SW Idle request (PMCSR<sub>y</sub>.REQSLP = 01<sub>B</sub>) issued by another CPU.</li> </ul> <p>The CPU code execution is halted and CPU clock is disabled in Idle state. The peripherals continue to remain active. CPU RAM memories (PSPR / DSPr / DLMU) are accessible to other bus masters and peripherals.</p> <p>CPU may exit Idle mode on following events:</p> <ul style="list-style-type: none"> <li>• When an interrupt occurs on a CPU returning the CPU to Run Mode.</li> <li>• When a trap occurs like an NMI trap event.</li> <li>• When the CPU watchdog or Safety watchdog timer overflow events trigger an SMU alarm in turn leading to a CPU interrupt.</li> <li>• When a MSB bit wrap of the CPU Watchdog counter takes place.</li> <li>• When a Application reset, System reset or any higher reset occurs.</li> <li>• On a SW Run request (PMCSR<sub>x</sub>.REQSLP = 00<sub>B</sub>) issued by another CPU.</li> </ul>

As shown in [Table 392](#), there are three main power modes available for the system:

- System Run Mode
- System Sleep Mode
- System Standby Mode

Furthermore, flexible reduction of power consumption is possible through following measures:

- Reduction of individual CPU power consumption by means of CPU clock scaling.
- Disabling the module clock by setting bit DISR in module CLC register if the module need not be active at the current point of time.
- Reducing the system frequency without changing individual peripheral clocks.
- Reducing individual peripheral clock frequency without changing system clock frequency. Main peripherals are provided with independent clocks separate from main system SRI and SPB clocks.

## Power Management System for Low-End (PMSLE)

**Table 392 System Power Management**

Mode	Description
<b>Run Mode</b>	At least one master CPU has not requested Sleep Mode or Standby mode and is in Run mode. All peripheral modules are active.
<b>Sleep Mode</b>	<p>System may enter Sleep Mode on following events:</p> <ul style="list-style-type: none"> <li>• On a SW Sleep request issued by setting PMCSR<sub>x</sub>.REQSLP = 10<sub>B</sub> by the master CPU. CPU code execution is halted and CPU Idle state is entered. Peripherals are set into sleep state if so configured in the respective CLCx.EDIS bit. Ports retain their earlier programmed state.</li> </ul> <p>System may exit Sleep mode on following events:</p> <ul style="list-style-type: none"> <li>• When an interrupt or trap occurs on the master CPU.</li> <li>• When an NMI trap event takes place.</li> <li>• When the CPU watchdog or Safety watchdog timer overflow events trigger an SMU alarm leading in turn to a master CPU interrupt.</li> <li>• When a MSB bit wrap of master CPU Watchdog counter takes place.</li> <li>• When an Application reset, System reset or any higher reset occurs.</li> </ul>

## Power Management System for Low-End (PMSLE)

**Table 392 System Power Management (cont'd)**

Mode	Description
Standby Mode (VEVRSB and VEXT supplied)	<p>System may enter Standby Mode on following events if so configured:</p> <ul style="list-style-type: none"> <li>• Standby entry on a SW Standby request issued by setting PMCSR<sub>x</sub>.REQSLP= 11<sub>B</sub> by the master CPU.</li> <li>• Standby entry on an ESR1 (NMI) assertion event. ESR1 (NMI) function doesn't require involvement of interrupt subsystem if configured as the standby entry trigger.</li> </ul> <p>The Standby domain constituting the Standby RAM, the 8 bit Standby Controller, shared ports and the wake-up unit remain actively supplied. The power to the rest of the chip is completely switched off. VEXT and VEVRSB rails remain supplied during Standby mode. VDDP3 and VDD supply rails are switched off.</p> <p>System may exit Standby mode on following events:</p> <ul style="list-style-type: none"> <li>• when a wake-up edge is detected on selected pins / ESR1.</li> <li>• when a wake-up edge is detected on ESR0.</li> <li>• when a wake-up request is issued by the 8 bit Standby Controller (SCR).</li> <li>• when a wake-up request is issued by Wake-up timer.</li> <li>• when PORST pin is asserted.</li> </ul>
Standby Mode (Only VEVRSB supplied)	<p>System may enter Standby Mode on following events if so configured:</p> <ul style="list-style-type: none"> <li>• Standby entry on a secondary under-voltage event during VEXT supply ramp-down.</li> <li>• Standby entry on an ESR1 (NMI) assertion event.</li> <li>• Standby entry on a SW Standby request issued by setting PMCSR<sub>x</sub>.REQSLP= 11<sub>B</sub> by the master CPU.</li> </ul> <p>The Standby domain constituting the Standby RAM and the 8 bit Standby Controller and Ports 33 and 34 remain actively supplied. The power to the rest of the chip is completely switched off. Only VEVRSB standby supply pin remain powered during Standby mode. VEXT, VDDP3 and VDD supply rails are switched off. SCR, WUT, Standby RAM supply maybe active or inactive during Standby mode.</p> <p>System may exit Standby mode on following event:</p> <ul style="list-style-type: none"> <li>• when VEXT supply ramps up</li> <li>• when a wake-up request is issued by SCR and VEXT is available.</li> <li>• when a wake-up request is issued by Wake-up timer and VEXT is available.</li> <li>• when a wake-up edge is detected on Pin B.</li> </ul>

## Power Management System for Low-End (PMSLE)

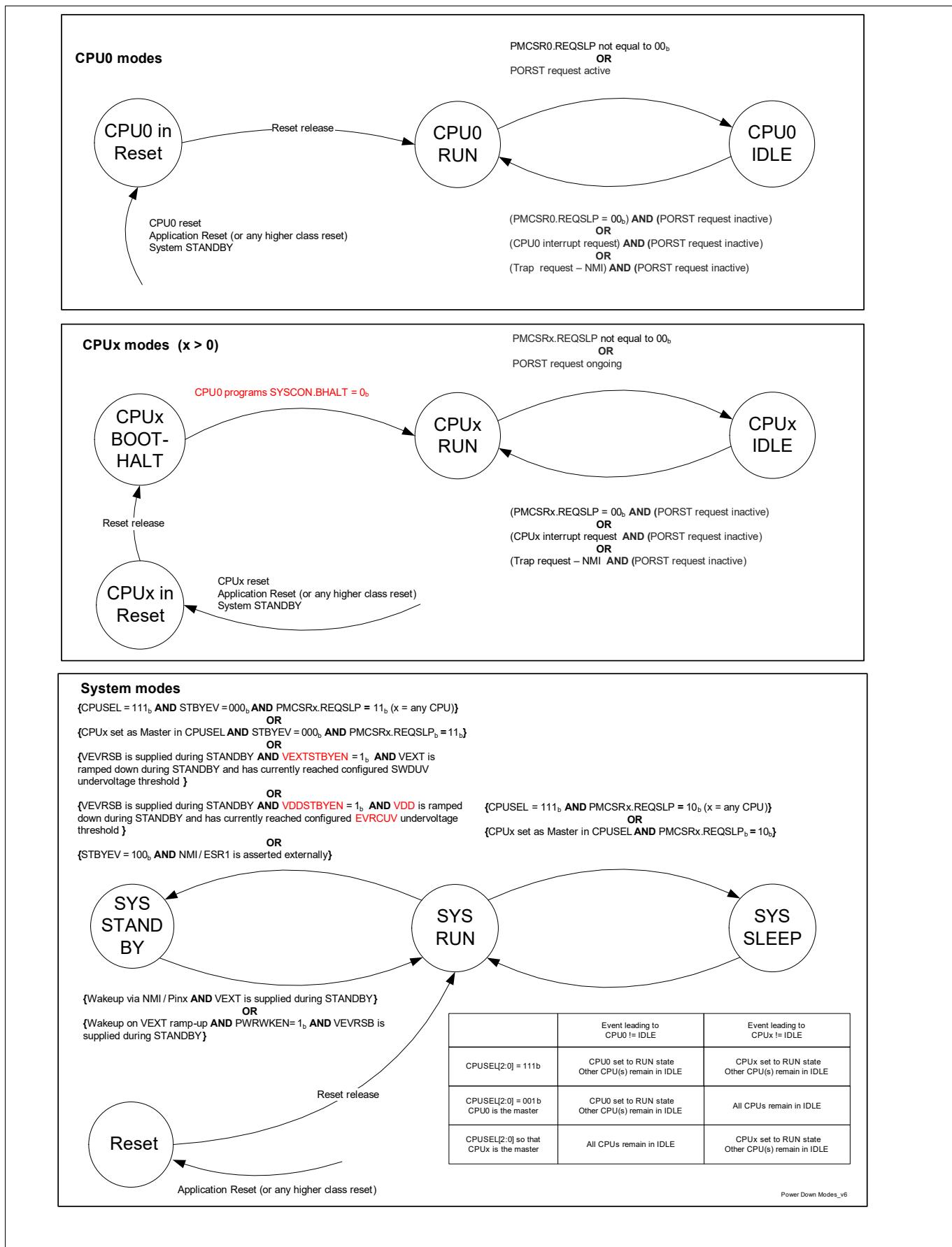


Figure 146 Power down modes and transitions

## Power Management System for Low-End (PMSLE)

### 12.2.3.2 Idle Mode

In case there are no active tasks to perform, a CPU may be requested to enter Idle mode during runtime by writing to the respective PMCSR<sub>x</sub> register and setting the bit field REQSLP = 01<sub>B</sub>.

#### 12.2.3.2.1 Entering Idle Mode :

Following events can invoke a CPU<sub>x</sub> Idle request

- CPU<sub>x</sub> setting itself in Idle by writing its own PMCSR<sub>x</sub> register: The respective PMCSR<sub>x</sub> register shall be accessed by setting CPU<sub>x</sub> ENDINIT = 0<sub>B</sub> and consequently writing the bit field REQSLP = 01<sub>B</sub>. The Idle transition takes place only when CPU<sub>x</sub> ENDINIT = 1<sub>B</sub> is set back again. This ensures that a CPU<sub>x</sub> does not enter Idle mode when it's WDTx is in Time-Out mode and ENDINIT<sub>x</sub> = 0<sub>B</sub> to avoid wake-up on a consequent WDT time-out. Safety ENDINIT mechanism shall not be used by a CPU to set itself into Idle to avoid wake-up on a Safety WDT time-out. Idle mode may also be simultaneously triggered for additional CPUs based on SCU\_PMSWCR1.CPUIDLSEL configuration.
- Masters except CPU<sub>x</sub> setting CPU<sub>x</sub> into Idle (e.g.- CPU[y]): The PMCSR<sub>x</sub> register shall be accessed by such masters by setting Safety ENDINIT = 0<sub>B</sub>. The Idle request is issued immediately on setting PMCSR<sub>x</sub>.REQSLP = 01<sub>B</sub>. The device no longer waits for Safety ENDINIT = 1<sub>B</sub> to be set back to trigger Idle transition. It need to be taken care to grant access via ACCEN register as required by application.

The CPU watchdog may be disabled or slowed down by reprogramming the timers before triggering Idle request via Software. On an Idle request, the CPU finishes its current operations and sends an acknowledge signal back to the Power Management unit. It then enters an inactive state in which the CPU clocks and the respective DMI and PMI memory units are shut off. It is recommended to reduce respective CPU clocks via CPU<sub>x</sub>DIV register bit field before issuing Idle request.

#### 12.2.3.2.2 State during Idle mode

During Idle Mode, memory accesses to the DMI, PMI and DLMU from other bus masters cause these units to wake-up automatically to handle these transactions. When memory transactions are complete, the DMI, PMI and DLMU return to Idle state again. Once Idle Mode is entered, the state is reflected in PMCSR<sub>x</sub>.PMST status bits.

**Table 393 CPU[x] Idle Mode Entry Sequence, Behavior and Status Indication**

Condition	CPU[x] writes PMCSR[x].REQSLP = 01 <sub>B</sub>	Masters except CPU <sub>x</sub> (e.g.- CPU[y]) writes PMCSR[x].REQSLP = 01 <sub>B</sub>	CPU[y] writes PMCSR[y].REQSLP = 01 <sub>B</sub>
CPU[x] enters Idle Mode	A CPU[x] should be able to set itself into Idle.  CE[x] = 0 <sub>B</sub> SE = 0 <sub>B</sub> or 1 <sub>B</sub> PMCSR[x].REQSLP= 01 <sub>B</sub> CPU[x] Idle Entry happens when CE[x] = 1 <sub>B</sub> is set. If CE[x] = 1 <sub>B</sub> during PMCSR[x] write; FPI error issued and request is not taken.	A CPU[y] or (other masters except CPU[x]) should be able to set another CPU[x] into Idle if it has SE rights.  SE = 0 <sub>B</sub> CE[x] = 0 <sub>B</sub> or 1 <sub>B</sub> PMCSR[x].REQSLP= 01 <sub>B</sub> CPU[x] Idle Entry happens immediately. If SE = 1 <sub>B</sub> during PMCSR[x] write; FPI error issued and request is not taken.	CPUIDLSEL = y+1 is already set by a CPU having SE rights before.  CE[y] = 0 <sub>B</sub> SE = 0 <sub>B</sub> or 1 <sub>B</sub> PMCSR[y].REQSLP= 01 <sub>B</sub> All CPUs go into IDLE. when CE[y] = 1 <sub>B</sub> is set. If CE[y] = 1 <sub>B</sub> during PMCSR[y] write; FPI error issued and request is not taken.

## Power Management System for Low-End (PMSLE)

**Table 393 CPU[x] Idle Mode Entry Sequence, Behavior and Status Indication (cont'd)**

Condition	CPU[x] writes PMCSR[x].REQSLP = 01 <sub>B</sub>	Masters except CPUx (e.g.- CPU[y]) writes PMCSR[x].REQSLP = 01 <sub>B</sub>	CPU[y] writes PMCSR[y].REQSLP = 01 <sub>B</sub>
CPU[x] during Idle Mode	PMCSR[x].REQSLP= 01 <sub>B</sub> PMCSR[x].PMST= 011 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>	PMCSR[x].REQSLP= 01 <sub>B</sub> PMCSR[x].PMST= 011 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>	PMCSR[ALL].REQSLP= 01 <sub>B</sub> PMCSR[ALL].PMST= 011 <sub>B</sub> PMSTAT0.CPU[ALL]&LS= 0 <sub>B</sub>
CPU[x] exits Idle mode	PMCSR[x].REQSLP= 00 <sub>B</sub> PMCSR[x].PMST= 001 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>	PMCSR[x].REQSLP= 00 <sub>B</sub> PMCSR[x].PMST= 001 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>	PMCSR[x].REQSLP= 00 <sub>B</sub> PMCSR[x].PMST= 001 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>

### 12.2.3.2.3 Exiting Idle mode

In Idle mode, the CPU will return to Run mode in response to the following wake-up events:

- An interrupt / trap received from an interrupt / trap source mapped to the CPU.
- An NMI trap request is received to wake-up the corresponding CPUs.
- A MSB bit wrap of the corresponding CPU Watchdog counter occurs.
- Setting the register bits PMCSR<sub>x</sub>.REQSLP = 00<sub>B</sub> to set the CPU<sub>x</sub> into Run mode.

The system enters reset state on an Application, System reset or any higher reset. If it is woken by a watchdog timer overflow event routed via the SMU to the CPU or by an NMI or by an interrupt, the CPU will immediately vector to the appropriate interrupt / trap handler.

CPU module reset will not result in exit from Idle mode if it was already in Idle state before. An explicit wake-up event has to happen before CPU is in run state again.

### 12.2.3.3 Sleep Mode

Sleep mode allows a progressive reduction of power consumption by gating the clocks of selected peripherals and keeping bare minimum modules active at their minimum clock frequencies during the Sleep state. Sleep mode maybe used to cater to Pretended Networking or ECU Degradation requirements. The clocks to a module maybe disabled individually using the respective CLCx.DISR register bits. Alternatively the clocks to selected peripherals may be simultaneously gated on a common sleep request if respective CLCx.EDIS register bits are cleared. The power consumption during Sleep state is predominantly dominated by the device leakage as power to the modules in core domain are not switched off. The dynamic core current component is reduced to the minimum as most of the module clocks are gated.

#### 12.2.3.3.1 Entering Sleep Mode

System may be requested to enter Sleep mode via software by master CPU by writing to the CPU's PMCSR<sub>x</sub> register and setting the bit field PMCSR<sub>x</sub>.REQSLP = 10<sub>B</sub>.

An example sequence for Sleep mode is enumerated below :

- The CLCx.EDIS register bit shall be cleared for all peripherals intended to be inactive in Sleep mode.
- All CPUs except the master CPU may be put into IDLE state. The respective watchdogs may be disabled or re-configured for slower modes. This allows to sequence the CPU load jumps before going into sleep mode
- Master CPU code execution maybe switched from Flash to PSPR RAM if required. Flash module may be explicitly set into Sleep state.
- The analog modules EVADC and EDSADC maybe switched off if not required to be active in Sleep state.
- It should be ensured to select the individual clocks from Clock Control Unit for peripherals which need to remain active during Sleep mode as shown in **Table 394**. Certain communication and timer peripherals have

## Power Management System for Low-End (PMSLE)

clocks independent from the system frequencies, namely SRI and SPB clocks, to allow the possibility to bypass the system PLL. In such cases, the System PLL is switched into bypass mode and consequently the DCO would be switched off. Peripheral clock will continue to run clocking the modules active during sleep mode. The system clock frequencies, namely SRI and SPB clocks, maybe then reduced to the minimum possible values via the low power divider and / or Kx divider to reduce the current consumption. In some cases the respective peripherals may be clocked directly from external crystal / resonator depending on application.

- The interrupt control unit provides the infrastructure for wake-up from sleep state and therefore need to be kept active with a minimum SPB bus frequency. The respective module wakeup interrupts are routed to master CPU to wake-up on an interrupt event.
- Sleep Mode may be entered as a unanimous decision of all the CPUs when ALL of the PMCSR<sub>x</sub> registers in the system request it AND SCU\_PMSWCR1.CPUSEL = 111<sub>B</sub>. Sleep Mode may also be entered based on a singular decision of a master CPU based on the configuration of the CPUSEL register. The PMCSR<sub>x</sub> register shall be accessed by setting CPU<sub>x</sub> ENDINIT = 0<sub>B</sub>. The Sleep request is issued only after CPU<sub>x</sub> ENDINIT bit is set back again. Safety ENDINIT mechanism shall not be used by a CPU<sub>x</sub> to issue Sleep request. The master CPU watchdog may also be disabled or slowed down before issuing a Sleep request.

### 12.2.3.3.2 State during Sleep Mode

Sleep Mode is disabled for a unit if CLCx.EDIS bit is set. The sleep request is ignored in this case and the corresponding unit continues normal operation as intended. If CLCx.EDIS is cleared, the clock of the module is gated. CPU Idle state is entered for all the CPUs as described in the previous section. All ports retain their earlier programmed state. The current consumption during Sleep mode is documented in datasheet.

**Table 394 Module activity and configuration during Sleep mode**

Module active during Sleep mode	Module and Clock State during Sleep Mode
MCAN	<p>Peripheral PLL active providing module clock (e.g. - f MOD = 20MHz - 40MHz).          Module may alternatively run on f OSC0 allowing also complete switch off of the Peripheral PLL.          Module FIFO and DMA allows autonomous handling of messages without involvement of CPU for a minimal amount of CAN messages.          System PLL may be switched into low power mode.          f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers (available only in B step).          System PLL may also be switched off and switched to Back-up clock depending on application (available only in B step).          Wake-up on CAN wake-up message identifier via CAN interrupt.</p>
ASCLIN	<p>Peripheral PLL active providing module clock (e.g. - f MOD = 20MHz).          Module may alternatively run on f OSC0 allowing also switch off of the Peripheral PLL.          System PLL may be switched into low power mode.          f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers.          Module FIFO and DMA allows autonomous handling of messages without involvement of CPU for a minimal amount of LIN frames.          System PLL may also be switched off and switched to Back-up clock depending on application.          Wake-up on LIN wake-up frame via ASCLIN interrupt.</p>

**Power Management System for Low-End (PMSLE)**
**Table 394 Module activity and configuration during Sleep mode (cont'd)**

<b>Module active during Sleep mode</b>	<b>Module and Clock State during Sleep Mode</b>
GPT12	<p>Peripheral PLL is disabled.</p> <p>System PLL may be switched into low power mode.</p> <p>f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers depending on application. Module clock (e.g - f MOD ~1-2 MHz) is derived from f SPB clock.</p> <p>System PLL may also be switched off and switched to Back-up clock depending on application.</p> <p>Wake-up on timer overflow or capture event via GPT12 interrupt.</p>
CCU6	<p>Peripheral PLL is disabled.</p> <p>System PLL may be switched into low power mode.</p> <p>f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers depending on application. Module clock (e.g - f MOD ~1-2 MHz) is derived from f SPB clock.</p> <p>System PLL may also be switched off and switched to Back-up clock depending on application.</p> <p>Wake-up on timer overflow or capture event via CCU6 interrupt.</p>
QSPI	<p>Peripheral PLL active providing module clock (e.g - f MOD = 20MHz).</p> <p>System PLL may be switched into low power mode.</p> <p>f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers.</p> <p>Services external watchdog if required by application. A timer module maybe used to trigger DMA or fill the FIFO allowing autonomous handling of messages without involvement of CPU.</p> <p>Wake-up in case of fault diagnosis of external device via a QSPI interrupt.</p>
Ethernet MAC	<p>Ethernet PHY active and provides module clock (e.g - f MOD = 25MHz) to the asynchronous part to decode the magic packet. Wake-up on magic packet via ETH interrupt.</p> <p>Alternatively PHY may trigger a wakeup directly via GPIO edge capture.</p> <p>Peripheral PLL may be disabled.</p> <p>System PLL may be switched into low power mode.</p> <p>f SRI, f SPB &amp; f ETH clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers.</p> <p>System PLL may also be switched off and switched to Back-up clock depending on application.</p>
I2C	<p>Peripheral PLL active providing module clock (e.g - f MOD = 20MHz).</p> <p>System PLL may be switched into low power mode.</p> <p>f SRI and f SPB clocks are reduced to (e.g. - 5 MHz) via LPDIV and / or Kx dividers.</p> <p>External communication remains active.</p>
GTM	<p>Peripheral PLL is disabled.</p> <p>System PLL is active and provides the module clock (e.g - f MOD ~f SPB ~ 1-2 MHz).</p> <p>f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers.</p> <p>It is recommended to switch off the GTM module completely and use smaller timer modules like CCU6 / GPT12, STM or WUT during Sleep state to reduce power consumption.</p> <p>Wake-up on timer overflow or capture event via GTM interrupt.</p>
STM	<p>Peripheral PLL is disabled.</p> <p>System PLL is active and provides the module clock (e.g - f MOD ~f SPB ~ 1-2 MHz).</p> <p>f SRI and f SPB clocks are reduced to (e.g. - ~1-2 MHz) via LPDIV and / or Kx dividers.</p> <p>Wake-up on timer overflow via STM interrupt.</p>

## Power Management System for Low-End (PMSLE)

**Table 394 Module activity and configuration during Sleep mode (cont'd)**

Module active during Sleep mode	Module and Clock State during Sleep Mode
Pin Wake-up ESR1 (NMI)	Peripheral PLL is disabled. System PLL bypassed. f SRI, f SRI clocks reduced to (e.g. - ~1-2 MHz) via LPDIV / Kx dividers. System PLL may also be switched off and switched to Back-up clock depending on application. Wake-up on Edge / Level detection on pin routed to ERUx, ESR1 (NMI) or PORT module via SCU interrupts or polling Port registers on an active timer interrupt.
WUT	Peripheral PLL is disabled. System PLL bypassed. f SRI, f SRI clocks reduced to (e.g. - ~1-2 MHz) via LPDIV / Kx dividers. System PLL may also be switched off as module clock is derived from Back-up clock Wake-up on timer overflow via WUT interrupt.

### 12.2.3.3.3 Exiting Sleep Mode

The system will exit Sleep mode on any wake-up event that causes any master CPU to exit Idle Mode depending on CPUSEL configuration. Only the master CPU associated with the interrupt wake-up event would be set into Run mode (REQSLP = RUN, PMST = RUN). Other CPUs will remain in Idle (REQSLP = SLEEP, PMST = IDLE). An NMI trap event will wake-up the respective CPU as configured in TRAPDIS0 and TRAPDIS1 registers. A MSB bit wrap of the corresponding master CPU Watchdog counter would also wake-up the master CPU. The response of the CPU to being woken up from Sleep Mode is also the same as for Idle Mode. Peripheral units that have entered Sleep Mode will switch back to their selected Run Mode operation. Wake-up latency from Sleep mode depends mainly on the extent of clock ramp-up required after wake-up keeping the load jump constraints. If DCO or PLL is switched off, the wake-up latency would include the time to power and lock the PLL. The sequence after wake-up is dependent on the entry sequence and mainly constitutes ramping back the clock system, activating analog and Flash modules, switching from RAM to Flash execution and activating additional CPUs. The time taken between interrupt trigger availability until CPU has woken up and is executing next instruction is less than 3 SPB + 20 SRI clock cycles.

**Power Management System for Low-End (PMSLE)**
**Table 395 System Sleep Mode Entry Sequence, Behavior and Status Indication**

<b>Condition</b>	<b>Master CPU[x] writes PMCSR[x].REQSLP = 10<sub>B</sub></b>	<b>CPU[y] writes PMCSR[x].REQSLP = 10<sub>B</sub></b>	<b>All CPU[y] writes respective PMCSR[y].REQSLP = 10<sub>B</sub></b>
System enters Sleep Mode	A CPUx should be able to trigger SLEEP mode if CPUSEL = x+1 <sub>B</sub> is already set by a CPU having SE rights before.  CE[x] = 0 <sub>B</sub> SE = 0 <sub>B</sub> or 1 <sub>B</sub> PMCSR[x].REQSLP= 10 <sub>B</sub> System enters SLEEP mode when CE[x] = 1 <sub>B</sub> is set. If CE[x] = 1 <sub>B</sub> during PMCSR[x] write; FPI error issued and request is not taken.	CPU[y] is not authorised to trigger Sleep Mode and therefore this is an error case.  CPUx is configured to trigger SLEEP mode via CPUSEL = x+1 <sub>B</sub> . SE = 0 <sub>B</sub> CE[x] = 0 <sub>B</sub> or 1 <sub>B</sub> PMCSR[x].REQSLP= 10 <sub>B</sub>	CPUSEL = 111 <sub>B</sub> is already set by a CPU having SE rights before. CE[y] = 0 <sub>B</sub> PMCSR[y].REQSLP= 10 <sub>B</sub> System enters SLEEP mode if all CPUs have requested for SLEEP entry and respective CE[y] = 1 <sub>B</sub> is set.  If CE[y] = 1 <sub>B</sub> during PMCSR[y] write; FPI error issued and request is not taken.
System during Sleep Mode	PMCSR[x].REQSLP= 10 <sub>B</sub> PMCSR[x].PMST= 100 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>  PMCSR[y].REQSLP= 10 <sub>B</sub> PMCSR[y].PMST= 011 <sub>B</sub> PMSTAT0.CPU[y] & LS= 0 <sub>B</sub>	PMCSR[x].REQSLP= 10 <sub>B</sub> PMCSR[x].PMST= 011 <sub>B</sub> PMSTAT0.CPU[x] & LS= 0 <sub>B</sub>  PMCSR[y].REQSLP= 01 <sub>B</sub> PMCSR[y].PMST= 011 <sub>B</sub> PMSTAT0.CPU[y] & LS= 0 <sub>B</sub>	PMCSR[y].REQSLP= 10 <sub>B</sub> PMCSR[y].PMST= 011 <sub>B</sub> PMSTAT0.CPU[y] & LS= 0 <sub>B</sub>
System during Sleep Exit	Wake-up on Master CPU PMCSR[x].REQSLP= 00 <sub>B</sub> PMCSR[x].PMST= 001 <sub>B</sub> PMSTAT0.CPU[x] & LS= 1 <sub>B</sub>  PMCSR[y].REQSLP= 10 <sub>B</sub> PMCSR[y].PMST= 011 <sub>B</sub> PMSTAT0.CPU[y] & LS= 0 <sub>B</sub>	System remains in RUN mode.	Wake-up event of respective CPU[x] PMCSR[x].REQSLP= 00 <sub>B</sub> PMCSR[x].PMST= 001 <sub>B</sub> PMSTAT0.CPU[x] & LS= 1 <sub>B</sub> Other CPU[y] remain in IDLE if not woken up PMCSR[y].REQSLP= 10 <sub>B</sub> PMCSR[y].PMST= 011 <sub>B</sub> PMSTAT0.CPU[y] & LS= 0 <sub>B</sub>

## Power Management System for Low-End (PMSLE)

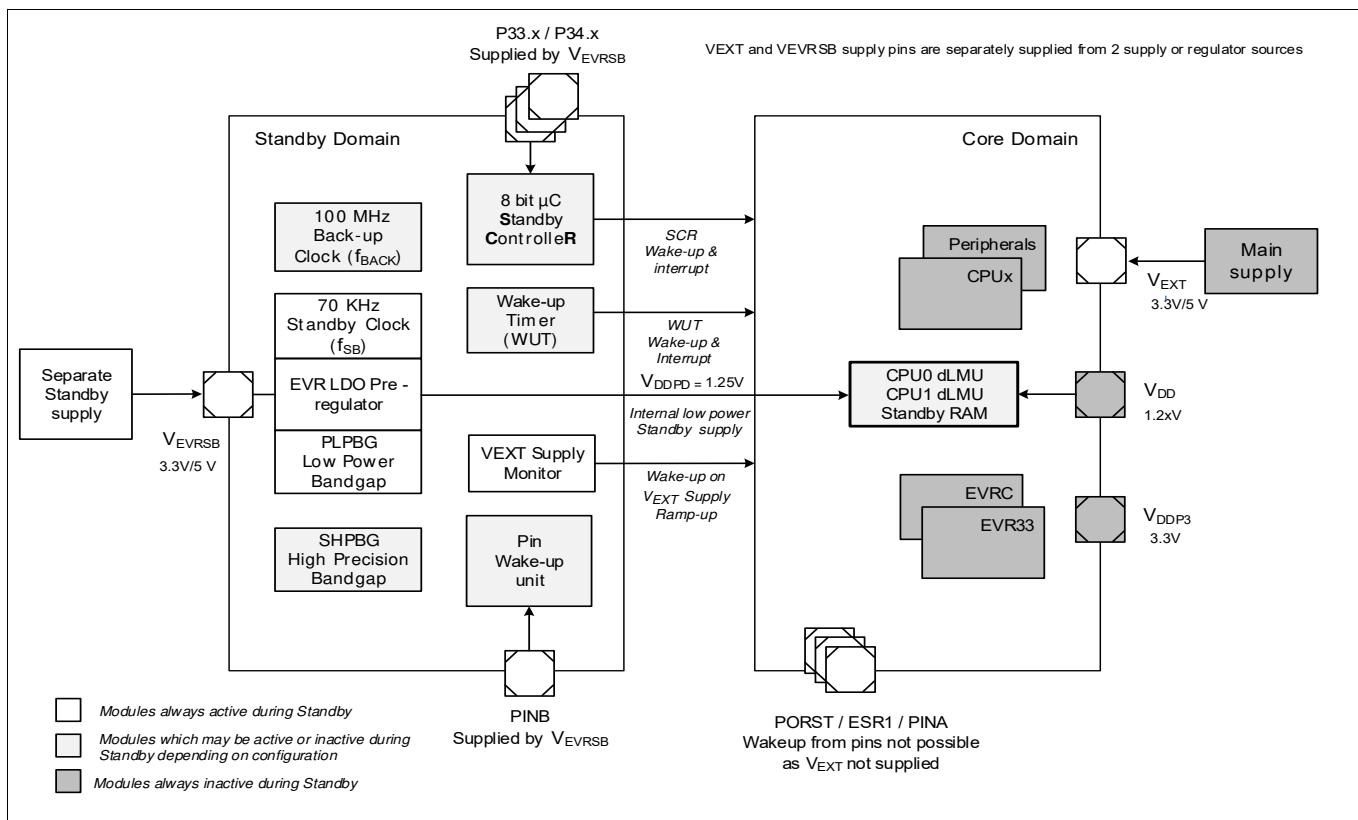
### 12.2.3.4 Standby Mode

The Standby domain constitutes the Standby RAM, the 8 bit Standby Controller, the Power Management unit, the Pin Wake-up unit, the Wake-up timer, the VEXT supply monitor and basic infrastructure components. The Standby domain is supplied by the EVRPR pre-regulator and is by default clocked by the 70 kHz internal low power clock source in Standby Mode. The 3.3V / 5V dedicated external Standby supply pin VEVRSB supplies the EVRPR pre-regulator and the Port domain P33.x / P34.x during the Standby mode when VEXT supply is switched off.

Following Standby topologies are supported with respective events which may trigger Standby mode entry and exit based on SCU\_PMSWCR1.STBYEV and **PMSWCR0.xWKEN** bits.

#### 12.2.3.4.1 Standby Mode with only VEVRSB domain supplied and VEXT domain switched off

As shown in **Figure 147**, only the Standby domain and Port domain P33.x/P34.x continue to be supplied by the separate VEVRSB supply pin during Standby mode. The main VEXT supply is switched off in Standby state. Consequently the rest of the PORT domain except P33.x/P34.x is devoid of supply.



**Figure 147 Standby domain supplied via a separate dedicated supply pin VEVRSB**

Standby Entry is triggered by following events :

- Standby entry on a secondary under-voltage event during VEXT supply ramp-down if configured in **PMSWCR0.VEXTSTBYEN** bits.
- Standby entry on SW request (PMCSR<sub>x</sub>.REQSLP = 11<sub>B</sub>) if configured via SCU\_PMSWCR1.STBYEV register bit field. The Standby request is issued only after CPU<sub>x</sub> ENDINIT bit is set back again. Safety ENDINIT mechanism shall not be used by a CPU<sub>x</sub> to issue Standby request. Standby entry on ESR1 (NMI) edge event if configured via SCU\_PMSWCR1.STBYEV register bit field.

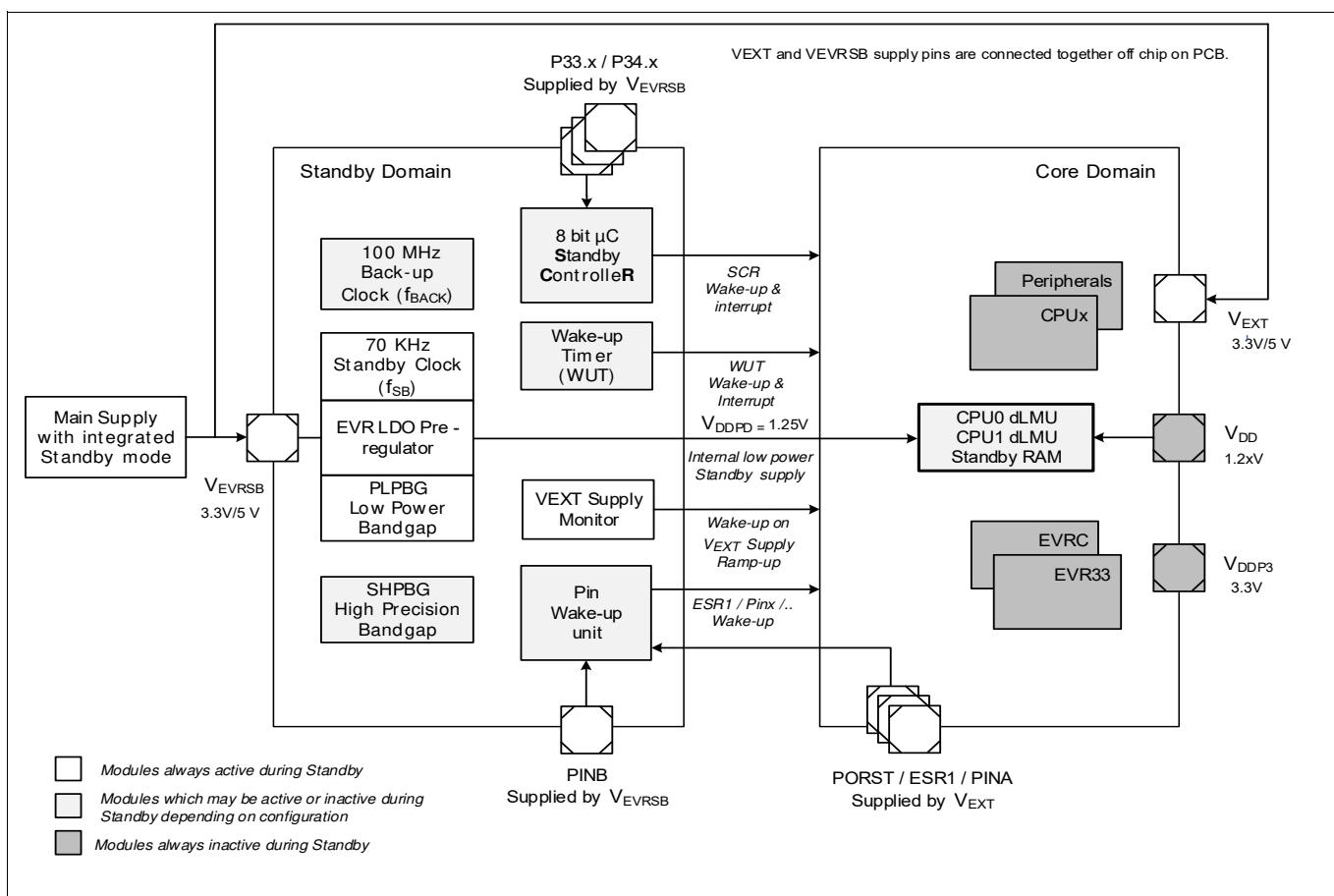
Standby Wake-up is triggered by following events after blanking filter time expiry :

## Power Management System for Low-End (PMSLE)

- Wake-up is triggered when main VEXT supply ramps-up again if configured via **PMSWCRO.PWRWKEN** enable bit.
- Wake-up is triggered by Standby Controller if configured via **PMSWCRO.SCRWKEN** enable bit provided VEXT supply has already ramped-up before. Standby controller can also request for VEXT ramp-up to external regulator.
- Wake-up via Wake-up Timer if configured via **PMSWCRO.WUTWKEN** enable bit provided VEXT has already ramped-up before.
- Wake-up via Pin B if configured via **PMSWCRO.PINBWKEN** enable bit provided VEXT has already ramped-up before.
- It is to be noted that wake-up via PORST, Pin A, ESR0 & ESR1 pins which are in turn supplied by VEXT is not supported during STANDBY as VEXT is not supplied. Therefore it is required to disable the respective **PMSWCRO.xWKEN** bits.

### 12.2.3.4.2 Standby Mode with both VEXT and VEVRSB supplied via common supply rail.

As shown in **Figure 148**, the Standby domain and the complete Pad domain including P33.x/P34.x, PORST, ESRx and PINx continue to be supplied by (VEVRSB + VEXT) supply rail during Standby mode. This allows additional wake-up possibility via PORST, ESRx and PINx pins but at the cost of higher power consumption during Standby mode.



**Figure 148 Standby domain supplied via a common supply rail connected to both VEXT and VEVRSB**

Standby Entry is triggered by following events

- Standby entry on a secondary under-voltage event during VEXT supply ramp-down if configured in **PMSWCRO.VEXTSTBYEN** bits.

## Power Management System for Low-End (PMSLE)

- Standby entry on SW request (PMCSR<sub>x</sub>.REQSLP = 11<sub>B</sub>) if configured via SCU\_PMSWCR1.STBYEV register bit field. The Standby request is issued only after CPU<sub>x</sub> ENDINIT bit is set back again. Safety ENDINIT mechanism shall not be used by a CPU<sub>x</sub> to issue Standby request.

- Standby entry on ESR1 (NMI) edge event if configured via SCU\_PMSWCR1.STBYEV register bit field.

Standby Wake-up is triggered by following events after blanking filter time expiry :

- Wake-up via NMI / Pinx: Wake-up on rising, falling or any edge of ESR1, Pin A or Pin B pins if configured via **PMSWCRO**.ESRxWKEN / PINxWKEN register bit fields.
- Wake-up is triggered by Standby Controller if configured via **PMSWCRO**.SCRWKEN enable bit.
- Wake-up via Wake-up Timer if configured via **PMSWCRO**.WUTWKEN enable bit.
- Wake-up via PORST pin if configured via **PMSWCRO**.PORSTWKEN enable bit.

### 12.2.3.4.3 Standby RAM

The Standby RAM constitutes ECC protected DLMU RAM of CPU0 (Block 0 and Block 1) and DLMU RAM of CPU1 (Block 0 and Block 1). The 32Kb Block 0 (lower half) is located at address \*0000H and 32Kb Block 1 (upper half) is located at address \*8000H of the respective address range of the corresponding CPUn DLMU RAM. The RAMs remain supplied during Standby mode if configured in **PMSWCRO**.STBYRAMSEL bits. On wake-up, the status which Standby RAMs remain supplied is reflected in **PMSWSTAT2**.STBYRAM bits. The initial 16 words from the start address of DLMU0/DLMU1 are not retained during standby mode as this memory region is used by start-up software.

The Standby RAM cell array is supplied by a separate supply pin (VEVRSB) during Standby state via the internal EVRPR Pre-Regulator. It shall be ensured that the external standby supply source continues to supply VEVRSB supply pin during Standby state with a supply between 2.6 V up to 5.5 V. Standby supply status is also monitored and indicated via RSTSTAT.STBYR bit which indicates EVRPR or VDDPD supply under-voltage LVD reset. It is to be taken care by the Start-up software after wake-up that Standby RAMs are not initialized if **PMSWSTAT2**.STBYRAM bits are set. Furthermore, if **PMSWCRO**.STBYRAMSEL bit is set to enable Standby RAM function and there was a VDD primary under-voltage (cold PORST) event, it is ensured that the Standby RAM supply is switched back to VDDPD supply rail. This ensures that RAM contents are not corrupted also during main VDD core supply loss.

### 12.2.3.4.4 VEXT Supply Monitor

If Standby mode is entered on a VEXT supply ramp down, the consequent wake-up on VEXT supply ramp up is triggered by the VEXT supply monitor activated by configuring **PMSWCRO**.PWRWKEN bit. The Standby request is issued by the secondary under-voltage monitor on crossing a voltage threshold as configured in **EVRUVMON** and **EVRMONCTRL** registers. Idle request acknowledge sequence issued to modules shall be deactivated on Standby entry by setting SCU\_PMSWCR1.IRADIS bit if VEXT supply is available. The EVR33 and EVRC regulators are switched off and Standby state is entered. Consequently VEXT and VDDM supplies may be ramped down, thus port and analog domains are also devoid of power. Wake-up is triggered when VEXT supply ramps up again and is detected by the VEXT supply monitor in the Standby domain. The detection time of the detector itself on reaching VLVDRST5 level is within 50 us. Nevertheless the complete time for start-up from Standby mode is quite the same as that for normal start-up and is documented tBP parameter in datasheet. VEXT wake-up is recognized as valid only after a minimum delay time has elapsed in Standby state as configured in **PMSWCRO**.BLNKFL register bits. This is to avoid spurious wake-up events owing to residual voltage on VEXT supply due to external buffer capacitors. After a successful wake-up, the register bit **PMSWSTAT**.PWRWKWP is set to indicate wake-up owing to a VEXT supply ramp-up and shall be cleared via **PMSWSTATCLR**.PWRWKPCCLR register bit.

### 12.2.3.4.5 Pin Wake-up Unit

External events may be mapped to ESRx / PINx pins in turn acting as wake-up signals for the system. In Run Mode, ESR1 pin may be used as fault or functional interface for external devices. In Standby Mode, an edge event on the

## Power Management System for Low-End (PMSLE)

ESR1 pin may be configured to trigger wake-up of the main core domain via **PMSWCRO**.ESR1WKEN bit and is reflected in **PMSWSTAT**.ESR1WKEN status flag. It can be configured to trigger a wake-up on rising, falling or both edges via **PMSWCRO**.ESR1EDCON bit. The minimum pulse width of the external wakeup input signal without the digital filter activated shall be atleast 2 clock cycles. Glitches on ESR1 input are filtered out by activating the filter via **PMSWCRO**.ESR1DFEN bit. The reset behavior is documented in External Service Requests chapter in RCU chapter. Additional pins (PINA - P14.1 and PINB - P33.12) may likewise be configured to trigger wake-up via **PMSWCRO**.xEDCON, xDFEN & xWKEN bits. On wake-up, **PMSWSTAT**.ESR1WKP or PINxWKP event flags provide information as to the wake-up source. It should be taken care after wake-up to clear the event flags via **PMSWSTATCLR** register. In case new wake-up events are captured while **PMSWSTAT**.xWKP flags are still set, then **PMSWSTAT**.xOVRRUN flags are set to indicate an overrun state owing to consecutive un-serviced wake-up events.

### 12.2.3.4.6 Standby ContolleR (SCR) Interface

The 8 bit Standby controller (SCR) subsystem constitutes an XC800 core, 8KB XRAM memory, various timer modules, ADC comparator, various communication peripherals and up to 16 shared pins executing autonomous activity during Idle, Sleep and Standby modes. Various Standby functions and periodic monitoring tasks may be encapsulated in the SCR with minimal power consumption overheads.

The SCR is enabled via **PMSWCR4**.SCREN bit and the status is reflected in **PMSWSTAT**.SCR bit. If SCR is disabled via **PMSWCR4**.SCREN bit, it is ensured that SCR 100MHz clock request, pending requests from SCR wake-up sources and other SCR interfaces do not have any effect on the main system. After start-up, CPU0 programs the SCR via FPI interface and copies the code into the internal XRAM.

A reset may be issued to the SCR via **PMSWCR4**.SCRSTREQ register bit. Consequently **PMSWSTAT**.SCRST status register bit is flagged to indicate SCR reset. SCRRST register bit is cleared via **PMSWSTATCLR**.SCRSTCLR bit. **PMSWCR4**.SCRSTREQ register bit is cleared after reset has been issued. The SCR is reset in case of warm PORST assertion based on **PMSWCR4**.PORSTREQ register configuration reflected in PORST register bit during normal RUN and SLEEP modes. The SCR is not affected by an Application or System reset. After reset release, the firmware initializes the SCR subsystem based on the hardware configuration programmed in **PMSWCR4**.SCRCFG bits.

The 20 MHz stand-by clock source is the default SCR clock active in System Standby Mode enabling higher-performance of the SCR subsystem. The SCR clock source may be switched to the internal 20 MHz (derived from the 100 MHz back-up clock) clock source via SCRCLKSEL register bit thus enabling higher performance on the SCR subsystem. Fundamentally SCR is in control of its clock needs and may request a clock switch via CMCON.OSCPD register overruling the behavior configured in SCRCLKSEL register. A watchdog ensures that the clock received after the request is adequate for reliable operation.

SCR PORT module shares a part of the PORT (P33.0 - P33.7, P33.9 - P33.15 and P34.1) domain with the main port system which may be kept active during Standby mode. The SCR ports are supplied by VDDPD and VEVRSB standby supply. The control to these pins need to be allocated explicitly to the SCR via port configuration Pxx\_PCSR register. Unused wake-up pins may be configured as tristate in Standby Mode. Furthermore dedicated wake-up pins, namely ESR0, ESR1, PINA - P14.1 and PINB - P33.12 are also routed to the SCR subsystem to recognise wake-up edges on these pins. When SCREN = 0 is programmed, PINB wakeup is configured as explained in [Section 12.2.3.4.5](#). Alternatively if SCREN=1 is programmed, PINB ownership is to be foremost transferred to SCR and SCR PINB Port configuration need to be set to input.

The SCR XRAM is accessible from the main domain via the FPI interface. Simultaneous access to XRAM via FPI interface and the SCR is arbitrated with SCR having default priority for XRAM access. In case of wake-up from Standby, it is ensured that SCR XRAM is not re-initialised.

An additional register interface with interrupt support using **PMSWCR2** register bit fields for exchange and facilitate status handshake between the two domains. The SCR can make a direct interrupt request to any CPUx by writing to register NMICON.SCRINTTC SCR register bit. An additional 8 bit information maybe written to

## Power Management System for Low-End (PMSLE)

SCRINTEXCHG SCR register which is also transferred to **PMSWCR2**.SCRINT register bit field to decode the interrupt reason. The routing of the interrupt to the service request node need to be enabled via **PMSIEN**.SCRINT register bit.

Likewise any CPU may also trigger a direct interrupt request to the SCR by writing to **PMSWCR2**.TCINTREQ register bit. An additional 8 bit information maybe written to **PMSWCR2**.TCINT register which is likewise transferred to TCINTEXCHG SCR register bit field to decode the interrupt reason on SCR side.

Critical SCR errors / events like XRAM ECC errors, SCR watchdog overflow event and SCR internal reset need to be communicated back to the main core domain via **PMSWCR2**.SCRECC, SCRWD and SCRRST register bits. These events may additionally trigger internal SCR reset if configured in RSTST SCR register. The occurrence of SCRECC, SCRWD and SCRRST events may be routed to interrupt based on **PMSIEN**.SCRECC, SCRWD and SCRRST register bits.

Like-wise resets of the main system, namely application, system and power-on resets, are reflected in **PMSWCR2**.RST register bit and communicated to SCR register MRSTST.RST bit. Furthermore, SMURST is differentiated via **PMSWCR2**.SMURST and communicated to SCR register MRSTST.SMURST bit. Interrupt maybe generated in SCR subsystem when MRSTST register bits are set. The bits are cleared when SCR has latched the information.

During standby mode, the SAR secondary monitor ADC maybe used to carry out analog conversions of up to 4 analog inputs (P33.4, P33.5, P33.6 & P33.7) if requested by SCR. Refer SCR ADCOMP chapter for more details.

During a standby to run mode transition on a wake-up event, the P33 and P34 PCSR.SELx shadow register value retains the programmed value of PCSR.SELx value before standby entry. This is to ensure that SCR continues to have control over the respective P33 and P34 port pins during and after exit from Standby, though the Port register PCSR.SELx value is reset. Only on a consequent explicit write to the register after reset release will a new PCSR.SELx value be taken to switch the Port 33 and P34 control.

The SCR may wake-up the main core domain from Standby state if configured in SCRWKEN register bit. The enabling of SCR wake-up via SCRWKEN should be programmed when SCR is running at 20 MHz. A wake-up request is issued by the SCR SW via SCRWP bit in STDBYWKP register as documented in the SCR SCU chapter. On wake-up of the main core domain, SCRWP event flag is set which shall be cleared via SCRWPCLR register bit.

### 12.2.3.4.7 Wake-up Timer (WUT)

The Wake-up Timer is a basic low power counter which may be used to wake-up the system periodically from Standby mode. The timer may also be used during RUN, IDLE or SLEEP modes. The following list enumerates the salient features.

- 24 bit counter running on 70 kHz clock source with programmable reload value.
- 24 bit counter status register providing the current count value.
- Timer resolution of 70 kHz or (70 kHz / 2<sup>10</sup>) configured via a clock divider.
  - 14.3 us resolution : 14.3 us - 240 s range ± 60% default tolerance
  - 14.3 ms resolution : 14.3 ms - 2.7 days range ± 60% default tolerance
- 2 operating modes :
  - Auto Reload mode - WUT is started and stopped via Software. Automatic reload on counter underflow and triggers a system wake-up.
  - Standby Auto Stop mode - Counter starts counting down from reload value on Standby entry. Counter stops on underflow and triggers a system wake-up.
- Events on WUT counter underflow
  - Interrupt request on SRC\_PMSx (WUT) interrupt node on counter underflow during RUN, IDLE or SLEEP mode.
  - Wake-up trigger on counter underflow during STANDBY mode.

## Power Management System for Low-End (PMSLE)

- Capture trigger on counter underflow to CCU60\_CC60IND, CCU61\_CC60IND and GTM (TIM 0.7) for trimming purpose.
- Over-run indication of consecutive un-serviced wake-up triggers

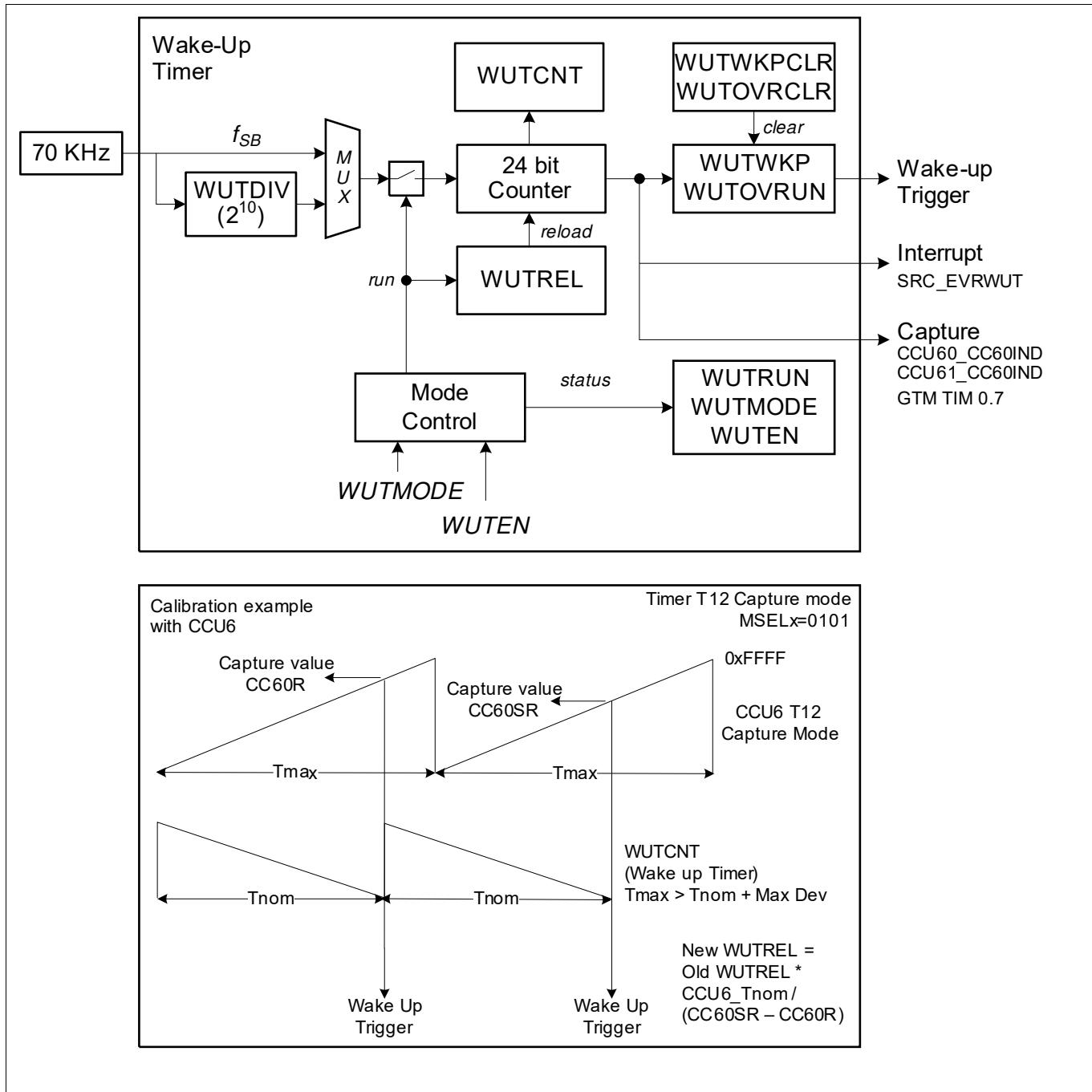


Figure 149 Wake-up Timer (WUT)

## Power Management System for Low-End (PMSLE)

**Table 396 Wake-up Timer Operation and Modes**

WUTEN	WUTMODE	Mode Description
0B	XB	<p>WUT is disabled and counter is stopped.</p> <p><b>PMSWCR3.WUTREL</b> reload value may be updated.</p> <p><b>PMSWSTAT2.WUTCNT,WUTRUN,WUTWKP &amp; WUTOVR</b> flags read 0.</p>
1B	0B	<p>Software Auto Reload mode :</p> <p>WUT starts running when <b>PMSWCR3.WUTEN</b> = 1 and <b>WUTMODE</b> = 0 is set. <b>PMSWCR0.WUTWKEN</b> = 1 is set to activate system wake-up from standby state. <b>PMSWSTAT.WUTRUN</b> bit is set indicating that WUT timer is currently running.</p> <p><b>PMSWUTCNT.WUTCNT</b> bit field indicates the actual counter value. On counter underflow, WUT is automatically reloaded with WUTREL value. During Standby, WUT underflow triggers system wake-up if <b>PMSWSTAT2.WUTWKEN</b> is set and <b>PMSWSTAT2.WUTWKP</b> flag is set. During Run, Idle or Sleep modes, WUT underflow triggers an interrupt request and <b>PMSWSTAT2.WUTWKP</b> flag is set. WUTREL reload value shall not be updated in this state.</p> <p>On wake-up, the <b>PMSWSTAT2.WUTWKP</b> flag shall be cleared by <b>PMSWSTATCLR.WUTWKPCLR</b> bit. In case of un-serviced consecutive counter underflow events, <b>PMSWSTAT2.WUTOVRUN</b> flag is set to indicate an over-run wake-up event.</p> <p>Interrupt over-run event can be detected via SRC.IOV bit during RUN mode.</p>
1B	1B	<p>Standby Auto Stop mode:</p> <p>The mode is selected by setting <b>PMSWCR3.WUTEN</b> = 1, <b>WUTMODE</b> = 1 is set. <b>PMSWCR0.WUTWKEN</b> = 1 is set to activate system wake-up from standby state.</p> <p>WUT starts running only when Standby mode is entered. On counter underflow, WUT stops running and the wake-up of system is triggered and <b>PMSWSTAT2.WUTWKP</b> flag is set.</p> <p>WUT starts running again on the next Standby mode entry. The intention is to have the timer running only during the Standby state.</p> <p><b>PMSWUTCNT.WUTCNT</b> eloads <b>PMSWCR3.WUTREL</b> value on a wake-up.</p> <p><b>PMSWSTAT.WUTRUN</b> reads 0 after a wake-up.</p> <p><b>PMSWCR3.WUTREL</b> reload value shall not be updated in this state.</p> <p>On wake-up, the <b>PMSWSTAT2.WUTWKP</b> flag shall be cleared by <b>PMSWSTATCLR.WUTWKPCLR</b> bit. In case system was woken up by other wake-up triggers while WUT was still running, an interrupt request is generated on WUT underflow. In case of un-serviced consecutive counter underflow events, <b>PMSWSTAT2.WUTOVRUN</b> flag is set to indicate an over-run wake-up event.</p> <p>Interrupt over-run event can be detected via SRC.IOV bit during RUN mode.</p>

In case of timer overflow, an interrupt is issued on the interrupt node SRC\_PMSx (WUT). Wake-up Timer reload value maybe trimmed during Run mode by comparing the time stamp on a WUT underflow captured by a GTM-TIM or CCU6x based on a more precise clock as shown in [Figure 149](#). This allows to compensate on short term the 70 kHz (fSB) clock source variations owing to technology, voltage and temperature.

## Power Management System for Low-End (PMSLE)

### 12.2.3.4.8 Entering Standby Mode (only VEVRSB domain supplied)

The Standby Mode entry may be requested via VEXT supply ramp-down triggered by a secondary SWDUV under-voltage event if configured in **PMSWCR0.VEXTSTBYEN** bits.

The Standby Mode entry may be requested by writing to PMCSR<sub>x</sub> register to set bit field REQSLP = 11<sub>B</sub> or via ESR1 (NMI) assertion as configured in SCU\_PMSWCR1.STBYEV bits.

Standby mode via SW may be entered based on a singular decision from a master CPU based on the configuration in the CPUSEL register. It may also be entered as a unanimous decision of all the CPUs when ALL of the PMCSR<sub>x</sub> registers in the system request it and PMSWCR1.CPUSEL = 111<sub>B</sub>. Each PMCSR<sub>x</sub> register is written by the corresponding CPU<sub>x</sub>.

Before entering Standby mode, modules may be sequentially shut off to avoid large load jumps.

- All peripherals and interrupts associated with CPUs except master CPU are switched off. This is to avoid wake-up of the CPUs once they are put into IDLE state.
- All CPUs except the master CPU are sequentially put into IDLE state. All watchdogs may be disabled or re-configured for slower modes. Peripherals module clocks are switched off in the respective CLC.DISR registers.
- Master CPU frequency reduction in steps compliant to load jump constraints. Master CPU code execution switched from Flash to PSPR RAM. Flash modules may be deactivated.
- System Clock is switched to internal 100 MHz clock source. System PLL & Peripheral PLL are switched off. Clock dividers are programmed to lower values.
- Standby SMU module shall be disabled via CMD\_STDBY.SMUEEN before going into Standby mode.
- SCU\_PMSWCR1.IRADIS bit set to disable Idle Request Acknowledge sequence activation for fast Standby Mode entry. System and Application reset generation possibilities are disabled. Only remaining reset possibility in this phase is via PORST or power-fail
- Standby RAM block selected via **PMSWCR0**.STBYRAMSEL bits. Dcache write back to be executed before Standby entry.
- Select the 70 kHz Standby clock source ( fSB ) via **PMSWCR4**.SCRCLKSEL bits. Configure the blanking filter appropriately via **PMSWCR0**.BLNKFIL bits.
- Configure pad state via **PMSWCR5**.TRISTREQ bit. All pads may be set into tristate or have pull-up device active. Regardless of the **PMSWCR5**.TRISTREQ setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate after standby mode entry. Configure **PMSWCR5**.ESR0TRIST bit to set ESR0 behavior as reset output active or tristate. In case of HWCFG [2:1,6] pins it is recommended to tie them to external pull devices.
- SCR may be kept running before entering the Standby state. The shared ports supplied by VEVRSB are configured either to be in tristate during standby or left to the control of SCR. The enabling of wake-up from SCR via **PMSWCR0**.SCRWKEN should be programmed when SCR is running at 20 MHz.
- Wake-up Timer may also be kept running before entering the Standby state. The wake-up from WUT may be activated via **PMSWCR0**.WUTWKEN bit.
- Wake-up via PORST, Pin A, ESR0 & ESR1 pins is not supported during Standby mode as VEXT will be ramped down on Standby entry. Therefore the respective **PMSWCR0**.PORSTWKEN, PINAWKEN, ESR0WKEN and ESR1WKEN wake-up configuration bits shall be disabled to avoid spurious wake-up triggers. It should be taken care that SCR is not reset on a standby entry by clearing **PMSWCR4**.PORSTREQ bit field.
- Enable wake-up on VEXT supply ramp-up via **PMSWCR0**.PWRWKEN bit. It needs to be ensured that both PWRWKEN and VEXTSTBYEN register bits are both set before entering Standby mode. VEXTSTBYEN register bitfield shall be set to ensure that when VEXT supply is removed during Standby state, no LVD reset is generated consequently exiting from Standby mode.
- In case Standby entry is triggered by VEXT supply ramp down, the threshold is configured in **EVRUVMON**.SWDUVVAL and transition condition in **EVRMONCTRL** register respectively. In case of nominal

## Power Management System for Low-End (PMSLE)

VEXT supply voltage of 5 V, it is recommended to configure SWDUVVAL register bitfield at 4 V for standby entry to have adequate distance to primary reset levels as well as operational region limits. In case of nominal VEXT supply voltage of 3.3V, it is recommended to configure SWDUVVAL register bitfield at 3.1 V for standby entry above primary reset levels. Nevertheless since there is only a minimal margin to reset levels in this case, Standby entry is additionally triggered by the crossing of primary undervoltage limits if VEXTSTBYEN register bit is set to ensure Standby entry in case of fast VEXT slopes. The parasitic diode path from VDDP3 to VEXT will keep the VEXT voltage at (VDDP3 - diode drop), so it should be ensured that SWDUVVAL is configured above (VDDP3 - diode drop) for standby entry if VDDP3 and VEXT supply rails are separately supplied. The selection of only VEXT supply voltage monitoring in **EVRMONCTRL** would reduce the secondary monitor standby entry latency time to (tMON/3). Configure Standby entry event in **PMSWCRO.VEXTSTBYEN** register bit.

- It shall be ensured that the primary under-voltage reset monitors are active before Standby entry is triggered and shall not be disabled in **EVRRSTCON** register.
- The external regulator is communicated to switch off VEXT supply. A controlled ramp-down of VEXT slope during Standby entry is recommended from external regulator (E.g - 0.5V/ms to 1.5V/ms). It need to be ensured that the VEXT supply is ramped below VEXT LVD reset level after standby entry before blanking filter time has expired. This is to avoid an immediate wake-up triggered by the residual VEXT voltage if it is above VEXT LVD reset level after blanking time has expired. It need to be also ensured that VDD and VDDP3 supply rails are consequently switched off after VEXT ramp down to reduce standby current within blanking filter time.
- All xWKP / xOVRUN flags activated by respective xWKEN bits shall be cleared before renewed Standby entry request, otherwise System will remain in Operation state and not enter Standby state. Standby request is issued via VEXT supply undervoltage event or via SW or NMI event. Once Standby entry event is recognised, the primary under-voltage reset generation is disabled and Standby RAM supply is switched from VDD to VDDPD within a single 25 MHz clock cycle. During Standby state entry, the wake-up logic is unable to detect wake-up events for a minimal time period less than 300 ns, therefore it need to be ensured that the wake-up pulse is asserted long enough that wake-up is detected. On entry into Standby mode, blanking filter is activated. The external standby regulator continues to supply the Standby domain via VEVRSB supply pin. Blanking filter is always activated on entry to Enter Standby state.
- Standby request issued via REQSLP bit field or ESR1/NMI event.
- Select required edge configuration in SCU\_ESRCFG1.EDCON if ESR1 is used as a trigger for standby entry.

## Power Management System for Low-End (PMSLE)

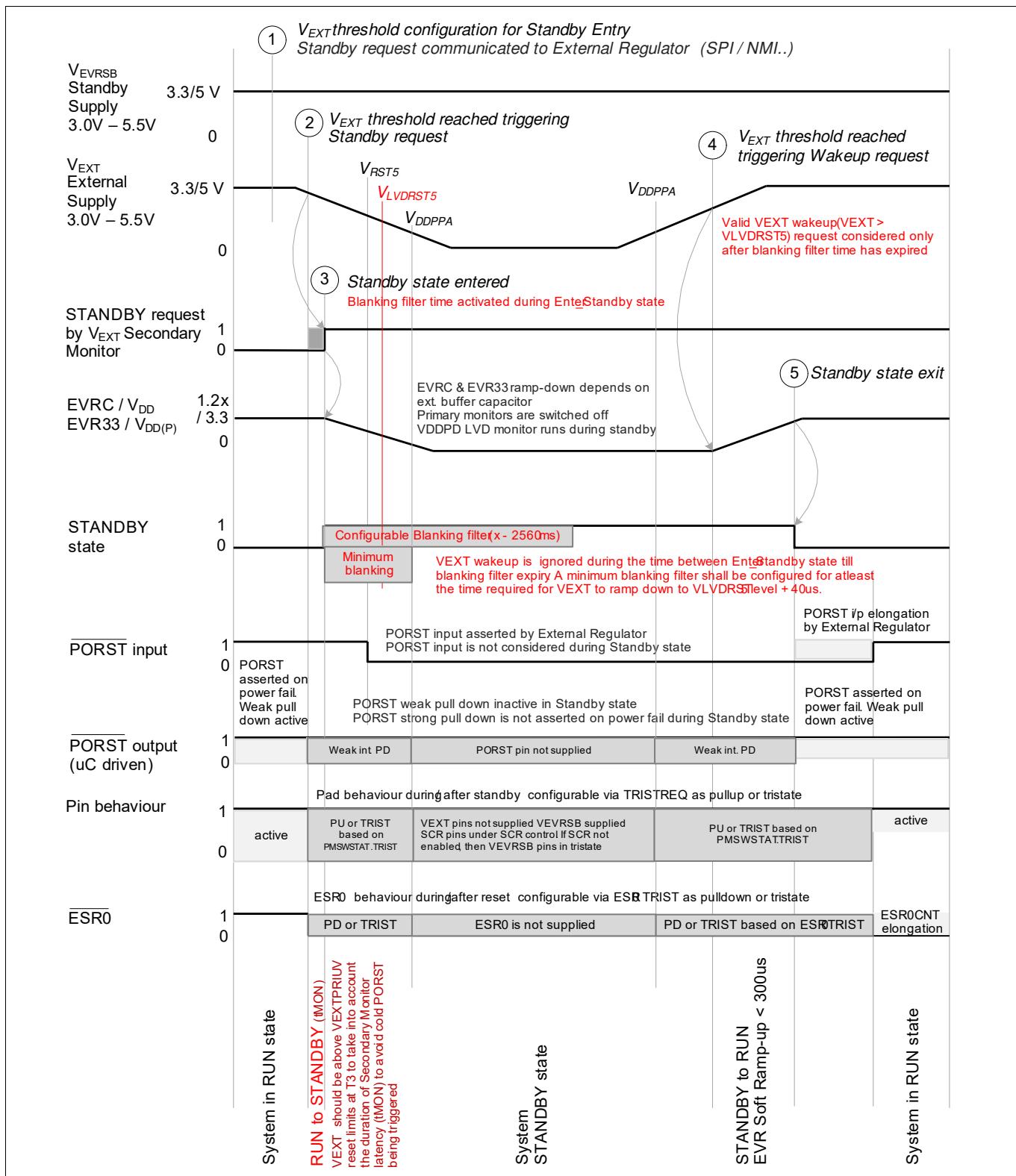


Figure 150 Standby entry on VEXT ramp-down and wake-up on VEXT ramp-up

## Power Management System for Low-End (PMSLE)

### 12.2.3.4.9 Entering Standby Mode (both VEVRSB and VEXT domain supplied)

The Standby Mode entry may be requested by writing to PMCSR<sub>x</sub> register to set bit field REQSLP = 11<sub>B</sub> or via ESR1 (NMI) assertion as configured in SCU\_PMSWCR1.STBYEV bits.

Standby mode via SW may be entered based on a singular decision from a master CPU based on the configuration in the CPUSEL register. It may also be entered as a unanimous decision of all the CPUs when ALL of the PMCSR<sub>x</sub> registers in the system request it and PMSWCR1.CPUSEL = 111<sub>B</sub>. Each PMCSR<sub>x</sub> register is written by the corresponding CPUx.

Before entering standby mode, various modules should be sequentially shut off in a sequence mainly to avoid large current jump on standby entry.

- All peripherals and interrupts associated with CPUs except master CPU are switched off. This is to avoid wake-up of the CPUs once they are put into IDLE state.
- All CPUs except the master CPU are sequentially put into IDLE state. CPU watchdogs may be disabled or re-configured for slower modes. Peripheral module clocks are switched off in respective CLC.DISR registers.
- Master CPU frequency reduction in steps compliant to load jump constraints. Master CPU code execution is switched from Flash to PSPR RAM. Flash modules are deactivated.
- System Clock is switched to the internal 100 MHz clock source. System PLL & Peripheral PLL are switched off. Clock dividers are programmed to lower values.
- Standby SMU module shall be disabled via CMD\_STDBY.SMUEN before going into Standby mode.
- Set SCU\_PMSWCR1.IRADIS bit to disable Idle Request Acknowledge sequence activation for fast Standby Mode entry. This ensures that standby request is not blocked by a pending reset request / sequence.
- Select the Standby RAM block via **PMSWCR0**.STBYRAMSEL bits. Dcache write back to be executed before Standby entry.
- Select the 70 kHz Standby clock source ( fSB ) via **PMSWCR4**.SCRCLKSEL bits. Configure the blanking filter appropriately via **PMSWCR0**.BLNKFIL bits.
- Select the clock source which need to be active on entry into Standby Mode via **PMSWCR4**.SCRCLKSEL bits. Wake-up trigger edge configuration and filter activation is configured via **PMSWCR0**.xxxEDCON and **PMSWCR0**.xxxDFEN bits.
- Configure pad state via **PMSWCR5**.TRISTREQ bit. All pads may either be in tristate or have pull-up devices active. Regardless of the **PMSWCR5**.TRISTREQ setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate after standby mode entry. Configure **PMSWCR5**.ESR0TRIST bit to configure ESR0 behavior as reset output or tristate during Standby and on wake-up. In case of HWCFG [2:1,6] pins it is recommended to tie them to external pull devices.
- Wake-up Timer may also be kept running before entering the Standby state. The wake-up from WUT may be activated via **PMSWCR0**.WUTWKEN bit.
- Enable ESR1 or PINx pins for wake-up via **PMSWCR0**.xxxWKEN bits.
- SCR may be kept running before entering the Standby state. The shared ports supplied by VEVRSB are configured either to be in tristate during standby or left to the control of SCR. The enabling of wake-up from SCR via **PMSWCR0**.SCRWKEN should be programmed when SCR is running at 20 MHz.
- All xWKP / xOVRUN flags activated by respective xWKEN bits shall be cleared before renewed Standby entry request, otherwise System will remain in Operation state and not enter Standby state. Configure Standby Entry event in SCU\_PMSWCR1.STBYEV register bits.
- It shall be ensured that the primary under-voltage reset monitors are active before Standby entry is triggered and shall not be disabled in **EVRRSTCON** register.
- Standby request issued via REQSLP bit field or ESR1/NMI event. An orderly shut down of various sub-systems is triggered to enter Standby mode. Once Standby entry request is recognised, the primary under-voltage

## Power Management System for Low-End (PMSLE)

reset generation is disabled and Standby RAM supply is switched from VDD to VDDPD within a single 25 MHz clock cycle. During Standby state entry, the wake-up logic is unable to detect wake-up events for a minimal time period less than 300 ns, therefore it need to be ensured that the wake-up pulse is asserted long enough that wake-up is detected. Blanking filter is always activated on entry to Enter\_Standby state. It need to be ensured that VDD and VDDP3 supply rails are consequently switched off after entry to standby to reduce standby current.

- Select required edge configuration in SCU\_ESRCFG1.EDCON if ESR1 is used as a trigger for standby entry.

### 12.2.3.4.10 State during Standby Mode

The Standby RAM (DLMU RAM of CPU0 and CPU1), the 8 bit Standby controller, the shared ports and the wake-up logic are kept alive in Standby mode. PORST pin, ESRx pins and PIN A provides wake-up function if VEXT is supplied. In case of wake-up on VEXT supply ramp-up and only VEVRSB is supplied, cold PORST function is resumed only after all supplies have ramped up.

All other pins are set in their default reset state. The default pin behavior during standby and after wake-up may be configured as pull-up or tristate accordingly by TRISTREQ register bit. Regardless of the [PMSWCR5.TRISTREQ](#) setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate after standby mode entry. All pins can be set into tristate except the TESTMODE pin where the internal pull-up is active also during Standby mode. The ESR0 pin may be configured as reset output or tristate during Standby mode by configuring ESR0TRIST bit. The shared port supplied by VEVRSB may retain their state. It need to be ensured by the external regulator that the VEVRSB voltage is within the operational region during Standby state.

The SCR continues to operate as a stand-alone 8 bit controller executing the intended operations. It should be ensured that the shared ports are configured in the corresponding port registers and the ownership of the pins are assigned either to the SCR or the main domain. In case the SCR needs to drive outputs during Standby mode, the default clock may need to be switched from 70 kHz to 20 MHz clock source. The SCR may request the EVR to activate or deactivate the 20 MHz clock. Analog conversions maybe carried out using SCR ADCOMP unit. SCR may be programmed to issue wake-up based on inputs from internal modules or shared pins. When the wake-up of the main core domain is required, the SCR issues a wake-up request via SCRWK bit in STDBYWKP register as documented in the SCR SCU.

### 12.2.3.4.11 Exiting Standby Mode - Wake-up event

The wake-up trigger in case of Standby mode where VEVRSB domain is only supplied may happen

- On a VEXT Supply ramp up after the blanking filter time has expired. SCR may be active. Wake-up can indirectly be triggered via SCR by communicating to external regulator to request the ramp-up of VEXT voltage. The wake-up reason could be any SCR event, Pin B edge transition or WUT Wake-up. Pin B edge transition or WUT Wake-up is also communicated to SCR as shown in [Figure 151](#).

After VEXT wakeup is recognised, it is expected that the VEXT supply is stable afterwards. In case of immediate VEXT powerfail consequent to VEXT wake-up, LVD reset or cold PORST may be triggered.

It is expected that the VEXT Supply has ramped down within the configured blanking filter time. Blanking filter shall be configured for atleast the time required for VEXT to ramp down to VLDRST5 level + 40 us. VEXT wakeup is ignored during the time between Enter Standby state till blanking filter expiry. A wake-up is triggered when the VEXT Supply is above the wake-up threshold of VLDRST5 for a time duration greater than 20 us indicated by event 4 in [Figure 150](#). Wake-up triggered on a VEXT Supply ramp-up is indicated in [PMSWSTAT2.PWRWKP](#) register bit and shall be cleared by [PMSWSTATCLR.PWRWKP](#) clear register bit.

The wake-up event in case of Standby mode where both VEVRSB and VEXT domain supplied may happen after the blanking filter time has expired on following events. ESRx / PINx edge, WUT underflow or SCR wakeup is ignored during the time between Enter\_Standby state till blanking filter expiry. xWKP / xOVRUN flags are only set during Standby mode after Blanking Filter expiry when the respective wake-up event happens.

## Power Management System for Low-End (PMSLE)

- ESR1 edge transition (NMI trap): **PMSWSTAT2.ESR1WKP** set on wake-up. **PMSWSTAT2.ESR1OVRUN** set to indicate overrun behavior in case of multiple un-serviced wake-ups.
- Pin A or Pin B edge transition (P14.1 or P33.12): **PMSWSTAT2.PINxWKP** set on wake-up. **PMSWSTAT2.PINxOVRUN** set to indicate overrun behavior in case of multiple un-serviced wake-ups.
- Wake-up from SCR via register STDBYWKP.SCRWKP in turn caused by following events. **PMSWSTAT2.SCRWKP** set on wake-up. **PMSWSTAT2.SCROVRUN** set to indicate overrun behavior in case of multiple un-serviced wake-ups.
  - Edge transitions at the shared ports
  - RTC interrupt
  - SCR watchdog overflow
  - Selected interrupts from communication modules
  - ADCOMP analog channel compare event.
- Wake-up from WUT

The main EVRC and EVR33 regulators are ramped up on wake-up based on the earlier latched configuration in **PMSWSTAT.HWCFCGEVR** register bits. On wake-up, all pads are either in tristate or are connected to pull-ups as indicated in **PMSWSTAT.TRIST** register bit. ESR0 behavior is indicated in **PMSWSTAT.ESR0TRIST** register bit. If Standby RAM was supplied during Standby state, it is indicated in **PMSWSTAT2.STBYRAM** register bits. Additional RAM integrity checks may be carried out after wake-up. RSTSTAT.STBYR bit indicates that the supply was reliable during Standby. The wake-up and over-run status flags are set in **PMSWSTAT2** register and shall be cleared by **PMSWSTATCLR** register. The wake-up time is nearly the same as the normal boot time as EVR need to be started and firmware need to be consequently executed.

## Power Management System for Low-End (PMSLE)

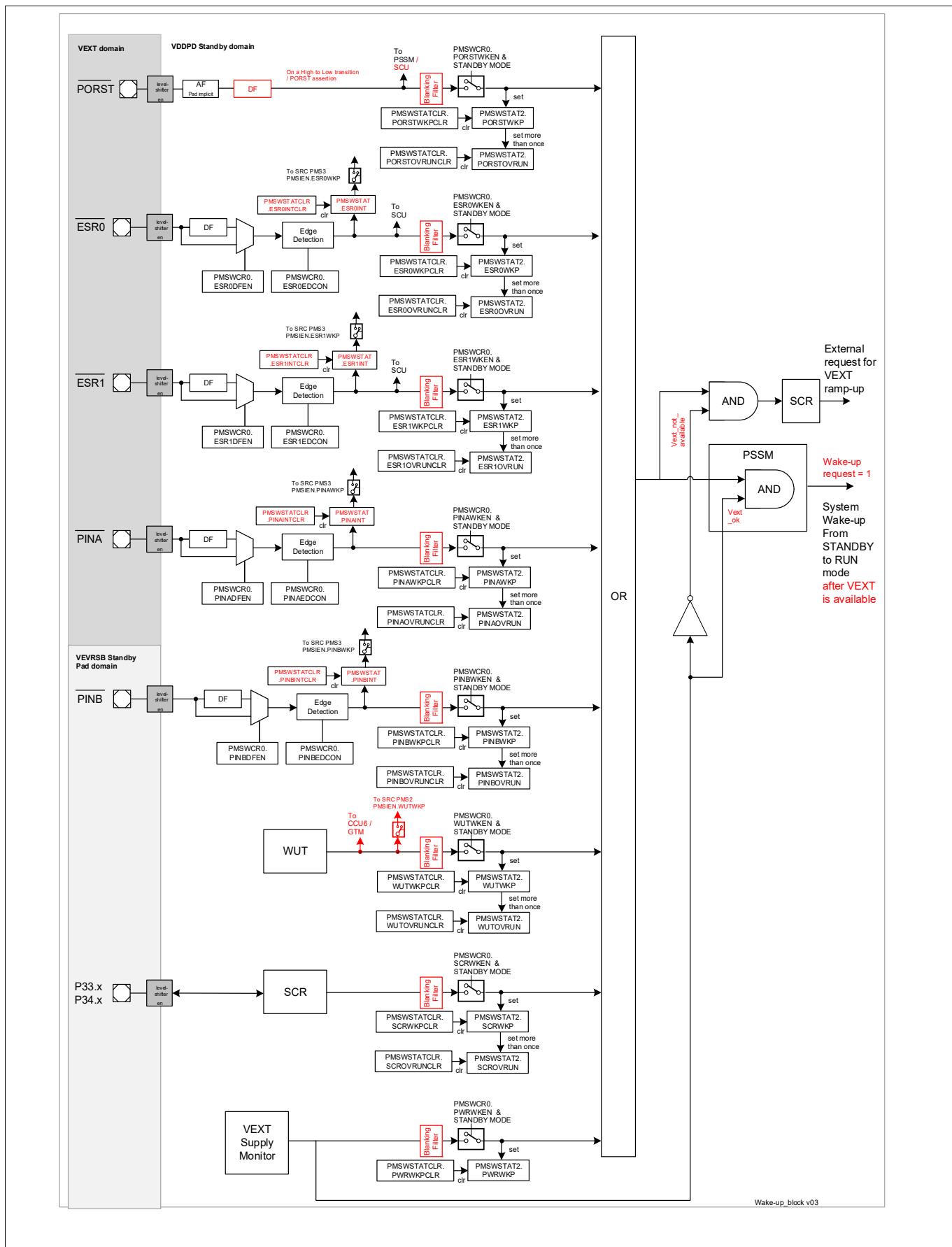


Figure 151 Wake-up Overview

## Power Management System for Low-End (PMSLE)

### 12.2.3.4.12 Exiting Standby Mode - Power Fail or Reset event

A power fail event of the Standby supply (VEVRSB pin) during Standby mode may inevitably result in the loss of Standby RAM contents. Consequently, LVD reset event is issued and the Standby domain is set into reset. EVR Pre-regulator under-voltage violation is indicated in RSTSTAT.STBYR flag which can be used as an indication whether Standby supply fail had happened and Standby RAM contents are reliable. Cold PORST flags RSTSTAT.SWD, EVRC and EVR33 would always be set after wake-up from STANDBY mode as these domains may be devoid of power during STANDBY to eliminate leakage current. It is recommended to keep a copy of the critical data also in Dflash in order to mitigate the effects if unwanted power fail events cannot be avoided.

In case of VEXT supply wake-up, PORST pin, ESRx pins and PIN A input are not evaluated during Standby mode as it is supplied by VEXT domain which is switched off during the STANDBY mode. In case VEXT domain is supplied during STANDBY mode, the Standby domain is woken up on PORST assertion depending on **PMSWCR0.PORSTWKEN** bit. **PMSWCR0.PORSTWKEN** is by default set to 1 to ensure wake-up on PORST assertion during STANDBY mode. The SCR may also set into reset simultaneously depending on **PMSWCR4.PORSTREQ** bit. The device boots up ramping up the regulators followed by firmware execution similar to a normal device start-up. On PORST wake-up, **PMSWSTAT2.PORSTWKP** event flag is set providing information as to the wake-up source. It should be taken care after wake-up to clear the event flags via **PMSWSTATCLR.PORSTWKPCCLR** register. In case new PORST wake-up events are captured while **PMSWSTAT2.PORSTWKP** flags are still set, then **PMSWSTAT2.PORSTOVRRUN** flags are set to indicate an overrun state owing to consecutive un-serviced wake-up events. The overrun flag is cleared via **PMSWSTATCLR.PORSTOVRUNCLR** bit.

The Standby RAM contents are kept intact after a wake-up caused by PORST assertion. In case of VDD supply under-voltage condition during wake-up phase, it is ensured that the Standby RAM is kept supplied by VDDPD until VDD is back in operational range to avoid Standby RAM data loss or corruption during transition. Reset is propagated to external devices via the ESR0 pin on exit from Standby mode depending on **PMSWCR5.ESR0TRIST** configuration. Firmware may elongate ESR0 reset output depending on Flash configuration.

Additional PORST digital filter activated via **PMSWCR5.PORSTDF** bit provides additional spike filtering of at least tPORSTDF duration to provide enhanced immunity against spurious spikes. This is in addition to the inherent analog PORST filter delay of the PORST pad / pin as documented in the datasheet. After cold PORST the additional PORST digital filter delay is by default inactive. If VEXT is supplied, PORST (high to low) during Standby state after blanking filter expiry triggers wake-up.

**Table 397 PORST pin assertion behavior on PMS and SCR subsystem during power modes**

Reaction to PORST pin assertion	RUN mode SLEEP mode	STANDBY mode
No reaction	No effect on PMS domain. No reaction on SCR if <b>PMSWCR4.PORSTREQ</b> = 0 but an SCR_NMI is triggered via the PMSWCR2.RST bit.	No effect on PMS domain if <b>PMSWCR0.PORSTWKEN</b> = 0. No reaction on SCR if <b>PMSWCR4.PORSTREQ</b> = 0
Wake-up	No effect on PMS or SCR domain as the system is already awake	PMS Standby to RUN transition takes place on PORST assertion when VEXT is supplied if <b>PMSWCR0.PORSTWKEN</b> =1(default)
Reset	No reset of PMS domain SCR is reset if <b>PMSWCR4.PORSTREQ</b> = 1 (default)	No reset of PMS domain SCR is reset if <b>PMSWCR4.PORSTREQ</b> = 1 (default)

## Power Management System for Low-End (PMSLE)

### 12.2.3.5 Load Jump Sequencing and Voltage Droop

Load jumps lead to consequent voltage overshoots / undershoots which need to be limited within the regulator dynamic specification and operational bounds of the supply rail. The initial phase after the load jump is buffered by the external capacitor which consequently leads to a linear discharge of the capacitor. Consequently the regulator feedback loop recognizes the deviation in voltage and reacts to the jump by changing the control output. The dimensioning of the capacitor results mainly from the load jump amplitude, ESR of the capacitor, the permissible voltage deviation and reaction time of the regulator. The capacitor size in turn has a tangible impact on BOM cost and PCB space. Minimizing peak-to-peak voltage deviation in the face of such large dynamic changes in load current need to be actively managed if large amounts of output capacitance are to be avoided.

If  $V_{DD}$  supply is generated by the internal EVRC regulator, the voltage transients owing to load jumps on core  $V_{DD}$  supply rail need to be restricted within  $V_{DD\_SETPOINT} + 8\% - 6\%$ . This includes a static accuracy of  $V_{DD\_SETPOINT} \pm 2\%$  and consequently  $+ 6\% - 4\%$  remaining for dynamic regulation.

In case of external  $V_{DD}$  supply, the voltage transients owing to load jumps on core  $V_{DD}$  supply rail need to be restricted within  $V_{DD\_SETPOINT} \pm 5\%$ . This includes a static accuracy of  $V_{DD\_SETPOINT} \pm 2\%$  and consequently  $\pm 3\%$  remaining for dynamic regulation.

Load jumps may be triggered by software or user driven actions or asynchronous hardware events. During software triggered non reset events, it is recommended to limit the load jumps ( $dIEXT/dt$ ,  $dIDD/dt$ ) to a maximum of 100 mA with 50 us settling time. For example, during clock ramp-up phase it is recommended to limit the clock switching steps so as not to violate this limit.

#### Handling load jump hardware events triggered asynchronously - Resets and NMI

In case of typical application load jump events, triggered asynchronously, like Non Maskable Interrupts and reset events, namely application, system and warm power-on reset requests, measures are built in to ensure that voltage overshoots are kept within bounds by load sequencing mechanisms or by means of register configuration.

In case of an NMI event, during RUN mode or waking up from SLEEP mode, the device may activate a large number of hitherto dormant circuits and wake-up the CPUs simultaneously resulting in a large change in load current. To avoid a large load jump on an NMI event, it needs to be ensured that only one CPU is triggered by the NMI or woken out of SLEEP mode and that other CPUs are still in IDLE mode. The other CPUs are consequently started one after another with adequate delay in between during start-up phase. The active CPU woken up on an NMI request is selected based on TRAPDIS0 and TRAPDIS1 register configurations.

In case of an Application Reset, System Reset or warm Power-On Reset request, the port pins are immediately set into reset state. Consequently the CPUs are ramped down in a sequence during the first 80 us immediately after the warm reset request. Finally after 180 us after reset request, the asynchronous reset event is issued to the device allowing to limit the maximum warm reset load jump to roughly half of the total dynamic IDD (IDDRAIL minus IDDPORST) current. It needs to be ensured that the VEXT, VDDP3 and VDD supply voltages are above the minimum operational voltage limits during the total reset phase of 180 us after warm reset request not to trigger a cold power-fail reset. Larger overshoots are tolerable during and after reset phase for a certain cumulated time but must be limited to operational and absolute maximum voltage ratings as documented in the datasheet.

Load jump events caused by asynchronous failure events like PLL loss of lock or external oscillator watchdog event may lead to overshoots which cannot be sequenced owing to the inherent nature of failure.

#### Handling simultaneous load jump requests triggered by Software

Software triggered Load Jump events include ramping up / down of various system clock frequencies, activating additional CPUs, Power mode transitions, CPU throttling and idle requests, MTU Memory tests, LBIST tests and so forth.

## Power Management System for Low-End (PMSLE)

In case of software triggered events, it is possible to prepare by lowering or raising the voltage setpoint before the load jump is issued. A negative voltage droop may be done before a negative load jump and a positive voltage droop before a positive load jump respectively. Thus negative load jumps leading to voltage overshoots is compensated partly by the negative voltage droop and likewise positive load jumps leading to voltage undershoots is compensated partly by the positive voltage droop as shown in [Figure 152](#). The voltage droop is configured through SCU\_PMTCSR0.SDSTEP register bits. The voltage droop in positive or negative direction is issued via SCU\_PMTCSR3.VDROOPREQ register bits. In case a current Vdroop request is not active or the Voltage Droop Timer is not currently running indicated via SCU\_PMTCSR3.VDTRUN or the Load Jump Timer is not currently running indicated via SCU\_PMTCSR2.LJTRUN, a new Vdroop request is taken. Once a new voltage droop request is issued, **EVRSTAT**.SDVOK is reset and TC3xx need to wait for a certain time till the regulator has settled on the new value which is realized using a Voltage Droop Timer. Once the regulator has settled on the new value, **EVRSTAT**.SDVOK status bit is set again indicating the end of the Voltage Droop transition and SCU\_PMTCSR3.VDTRUN and SCU\_PMTCSR3.VDTCNT is reset by hardware. If SDVOK status bit is set by EVRC before compare match of VDT has occurred, VDTOV overflow bit is not set and overrun interrupt is not generated. The Voltage Droop Timer compare value is configured in SCU\_PMTCSR1.VDTCV register bits and the current value is indicated in SCU\_PMTCSR3.VDTCNT register bits. In case of a compare match, the overflow SCU\_PMTCSR3.VDTOV bit is set if enabled via SCU\_PMTCSR0.VDTOVEN register bits. In this case, overflow bit has to be explicitly cleared via SCU\_PMTCSR3.VDTOVCLR before a new request can be taken to support a sequential polling based approach. Furthermore, interrupt maybe activated on an overflow if SCU\_PMTCSR0.VDTOVIEN is enabled.

Simultaneous software triggered load jump events can be likewise avoided by checking whether a Load Jump is ongoing or the Load Jump Timer is currently running. The Load Jump Request is issued by triggering a compare and swap operation on SCU\_PMTCSR2 register. A CPU will access data from SCU\_PMTCSR2 register and will compare the current value with an expected value. The expected value is that there is no load jump currently ongoing and VDTRUN bit state is 0. If there is a match, the CPU will make the swap by setting SCU\_PMTCSR2.LDJMPREQ variable and starting the timer. Obviously if multiple CPUs are making this operation simultaneously only one CPU will succeed and others will fail the compare and swap operation when the Load Jump Timer would be running. The idea is to prevent multiple CPUs from doing load jumps simultaneously by treating Load Jump as a critical section and ensuring that only a single CPU can check and issue a load jump request atomically. Once a load jump request is taken, a timer is started and other CPUs have to wait till the regulator output has been restored to the setpoint value and ensures adequate regulator reaction time. The other CPUs are not blocked during the waiting period instead they can continue with some other operations or try to make the request again at a later point of time.

Thus negative load jumps and positive load jumps are followed by a blanking period using a Load Jump Timer as shown in [Figure 152](#). The Load Jump Timer is configured through SCU\_PMTCSR0 register. The load jump request is issued via SCU\_PMTCSR2.LDJMPREQ register bits. If a current Load Jump request is not active or the Load Jump Timer is not currently running indicated via SCU\_PMTCSR2.LJTRUN or the Voltage Droop Timer is not currently running indicated via SCU\_PMTCSR3.VDTRUN, a new Load Jump request is taken. Once a new Load Jump request is issued, the device needs to wait for a certain time till the regulator has reacted to the jump which is realized using a Load Jump Timer. The Load Jump Timer compare value is configured in SCU\_PMTCSR1.LJTCV register bits and the current value is indicated in SCU\_PMTCSR2.LJTCNT register bits. In case of a compare match, the overflow SCU\_PMTCSR2.LJTOV bit is set if enabled via SCU\_PMTCSR0.LJTOVEN register bits. In this case, overflow bit has to be explicitly cleared via SCU\_PMTCSR2.LJTOVCLR before a new request can be taken to support a sequential polling based approach. Furthermore, an interrupt maybe activated on an overflow if SCU\_PMTCSR0.LJTOVIEN is enabled. Overflow bit is routed to an OS interrupt to schedule the next current jump. Overflow bit maybe masked or used in the compare and swap operation if SCU\_PMTCSR0.LJTOVEN is set to ensure that explicit clear of the time out has happened before issuing a new request. SCU\_PMTCSR2.LJTOVCLR also clears SCU\_PMTCSR3.VDROOPREQ and SCU\_PMTCSR2.LDJMPREQ request.

## Power Management System for Low-End (PMSLE)

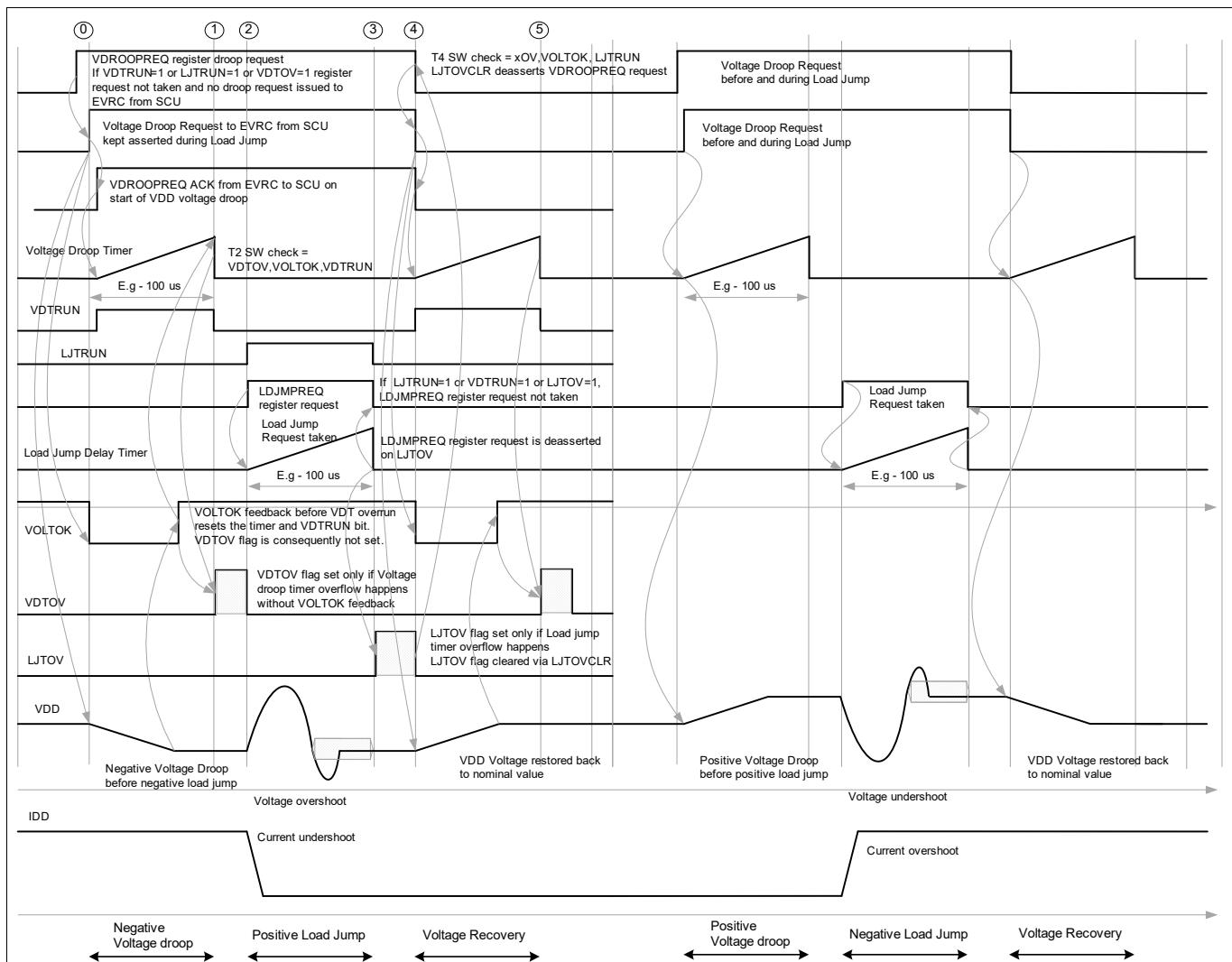


Figure 152 Load jumps and Voltage Droop

---

**Power Management System for Low-End (PMSLE)**

### 12.2.3.6 Core Die Temperature Sensor (DTSC)

The Core Die Temperature Sensor (DTSC) generates a measurement result that indicates directly the current temperature. The DTSC measures the temperature with an accuracy within ( $T_{NL} + T_{CALACC}$ ) parameter limits within the TSR temperature range documented in the datasheet. The result of the measurement is updated periodically in DTSCSTAT.RESULT register bit field with a resolution less than 1/5th of a degree Kelvin. The Die Temperature Sensor is available after an application reset release on a device start-up and temperature measurements are carried out continuously during normal RUN / SLEEP modes once DTSC is enabled. The Die Temperature Sensor and DTSLIM and DTSCSTAT registers are reset on an application reset.

The DTSC is enabled via DTSLIM.DTSEN register bitfield. The DTS start-up is completed after a nominal 20us delay after DTSLIM.DTSEN is set. After an ongoing temperature measurement is completed, DTSCSTAT.RESULT bit field is updated coherently with the new value. An interrupt service request (SRC\_SCUERU3) can be generated after a measurement is completed. DTS bandgap status is reflected in DTSLIM.BGPOK status flag. The DTS accuracy and measurement time is defined in the Data Sheet. The DTSLIM register shall be updated before enabling DTSC via DTSLIM.DTSEN register bitfield.

Die temperature upper and lower limits are configured in DTSLIM.UPPER and LOWER register bits. On violation of these limits, DTSLIM.UOF and LLU status bits are set and alarms are forwarded to core SMU. After start-up or application reset, the DTSC limits have to be re-configured appropriately depending on the application before alarm reactions from SMU are activated. Only when a new DTSC conversion result is available, the DTSC comparators are consequently triggered to check the actual DTSCSTAT.RESULT against the upper and lower limits.

DTSCCON and DTSCBGOCTRL register can be also changed after DTSC is enabled on the fly for test purposes.

## Power Management System for Low-End (PMSLE)

### 12.3 Registers

#### 12.3.1 Register Access Modes

Read and write access to registers and memory locations are sometimes restricted. In memory and register access tables, the terms as defined in **Table 398** are used.

In general, if an access type is not permitted under these rules (e.g. attempted write to R, attempted user mode access to SV, attempted access to E without Endinit, etc.) then a Bus Error will result, unless the access is also marked as nBE (or otherwise stated in the specific module chapter).

**Table 398 Access Terms**

Symbol	Description
BE	Always returns Bus Error (e.g. used on Write Access to indicate a read-only register). Bus error is captured by BCU which in turn generates alarm to SMU.
CEx	Access only when CPUx Endinit is not active (SCU_WDTCPUxCON0.ENDINIT = 0)
E	Access only when any Endinit is inactive (SCU_WDTCPUxCON0.ENDINIT = 0 for any CPUx, or SCU_EICON0.ENDINIT = 0)
P	Access only from Master x (when MOD_ACCEN0.ENx = 1). In case of access violation, bus error is generated which is captured by BCU which in turn generates alarm to SMU.
P0	Access only from Master x (when MOD_ACCEN00.ENx = 1)
P1	Access only from Master x (when MOD_ACCEN10.ENx = 1)
PW	Access only when correct Password
SE	Access only when Safety Endinit is inactive (SCU_WDTSICON0.ENDINIT = 0 or SCU_SEICON.ENDINIT = 0). In case of access with Safety Endinit active, bus error is generated which is captured by BCU which in turn generates alarm to SMU.
ST	Access only when startup (SSW) executes and writable only while bit STCON.STP is cleared.
SV	Access only when PSW = Supervisor Mode. In case of access violation, bus error is generated which is captured by BCU which in turn generates alarm to SMU.
TM	Access only when SCU test mode
U	Access only when PSW = User Mode 0 or 1 Reset Value: Value or bit is not changed by a reset operation.
32	Access only when 32-bit width
32/16	Access only when 32-bit or 16-bit width

**Table 399 Combined Access Conditions**

Symbol	Description
P, U, SV	P and (U or SV)
U, SV	(U or SV)
P, SV, E	P and SV and E
SV, SE	SV and SE
TM, ST	TM or ST

---

**Power Management System for Low-End (PMSLE)**
**Table 400 Other Register Annotations**

<b>Symbol</b>	<b>Description</b>
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.
nE	Indicates that no Error is generated when accessing this address or address range, even though the access is to an undefined address or address range. True for CPU accesses (MTCR/MFCR) to undefined addresses in the CSFR range.

## Power Management System for Low-End (PMSLE)

### 12.3.2 Power Management Control Registers (PMS)

**Table 401 Register Address Space - PMS**

Module	Base Address	End Address	Note
(sx_fpi)	F0240000 <sub>H</sub>	F0241FFF <sub>H</sub>	
sx_fpi	F0248000 <sub>H</sub>	F02481FF <sub>H</sub>	FPI slave interface

**Table 402 Register Overview - PMS (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
ID	Identification Register	0008 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">83</a>
EVRSTAT	EVR Status Register	002C <sub>H</sub>	U,SV	BE	See page <a href="#">84</a>	<a href="#">84</a>
EVRADCSTAT	EVR Primary ADC Status Register	0034 <sub>H</sub>	U,SV	BE	LVD Reset	<a href="#">89</a>
EVRRSTCON	EVR Reset Control Register	003C <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">90</a>	<a href="#">90</a>
EVRRSTSTAT	EVR Reset Status Register	0044 <sub>H</sub>	U,SV	BE	See page <a href="#">93</a>	<a href="#">93</a>
EVRTRIM	EVR Trim Control Register	004C <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">94</a>	<a href="#">94</a>
EVRTRIMSTAT	EVR Trim Status Register	0050 <sub>H</sub>	U,SV	BE	See page <a href="#">96</a>	<a href="#">96</a>
EVRMONSTAT1	EVR Secondary ADC Status Register 1	0060 <sub>H</sub>	U,SV	BE	See page <a href="#">97</a>	<a href="#">97</a>
EVRMONSTAT2	EVR Secondary ADC Status Register 2	0064 <sub>H</sub>	U,SV	BE	See page <a href="#">99</a>	<a href="#">99</a>
EVRMONCTRL	EVR Secondary Monitor Control Register	0068 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">100</a>	<a href="#">100</a>
EVRMONFILT	EVR Secondary Monitor Filter Register	0070 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">105</a>	<a href="#">105</a>
PMSIEN	PMS Interrupt Enable Register	0074 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">107</a>	<a href="#">107</a>
EVRUVMON	EVR Secondary Under-voltage Monitor Register	0078 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">110</a>	<a href="#">110</a>
EVROVMON	EVR Secondary Over-voltage Monitor Register	007C <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">111</a>	<a href="#">111</a>
EVRUVMON2	EVR Secondary Under-voltage Monitor Register 2	0080 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">113</a>	<a href="#">113</a>
EVROVMON2	EVR Secondary Over-voltage Monitor Register 2	0084 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">114</a>	<a href="#">114</a>
HSMUVMON	EVR Primary HSM Under-voltage Monitor Register	0088 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">115</a>	<a href="#">115</a>
HSMOVMON	EVR Primary HSM Over-voltage Monitor Register	008C <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">117</a>	<a href="#">117</a>
EVR33CON	EVR33 Control Register	0090 <sub>H</sub>	U,SV	SV,SE,P	See page <a href="#">119</a>	<a href="#">119</a>

## Power Management System for Low-End (PMSLE)

Table 402 Register Overview - PMS (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
EVROSCCTRL	EVR Oscillator Control Register	00A0 <sub>H</sub>	U,SV	SV,SE,P	See page 121	121
PMSWCR0	Standby and Wake-up Control Register 0	00B4 <sub>H</sub>	U,SV	SV,SE,P	LVD Reset	149
PMSWCR2	Standby and Wake-up Control Register 2	00B8 <sub>H</sub>	U,SV	SV,SE,P	LVD Reset	153
PMSWCR3	Standby and Wake-up Control Register 3	00C0 <sub>H</sub>	U,SV	SV,SE,P	LVD Reset	154
PMSWCR4	Standby and Wake-up Control Register 4	00C4 <sub>H</sub>	U,SV	SV,SE,P	LVD Reset	155
PMSWCR5	Standby and Wake-up Control Register 5	00C8 <sub>H</sub>	U,SV	SV,SE,P	LVD Reset	157
PMSWSTAT	Standby and Wake-up Status Register	00D4 <sub>H</sub>	U,SV	BE	LVD Reset	159
PMSWSTAT2	Standby and Wake-up Status Register 2	00D8 <sub>H</sub>	U,SV	BE	LVD Reset	162
PMSWUTCNT	Standby WUT Counter Register	00DC <sub>H</sub>	U,SV	BE	LVD Reset	159
PMSWSTATCLR	Standby and Wake-up Status Clear Register	00E8 <sub>H</sub>	U,SV	SV,SE,P	LVD Reset	168
EVRSDDSTAT0	EVR SD Status Register 0	00FC <sub>H</sub>	U,SV	BE	See page 122	122
EVRSDDCTRL0	EVRC SD Control Register 0	0108 <sub>H</sub>	U,SV	SV,SE,P	See page 123	123
EVRSDDCTRL1	EVRC SD Control Register 1	010C <sub>H</sub>	U,SV	SV,SE,P	See page 124	124
EVRSDDCTRL2	EVRC SD Control Register 2	0110 <sub>H</sub>	U,SV	SV,SE,P	See page 126	126
EVRSDDCTRL3	EVRC SD Control Register 3	0114 <sub>H</sub>	U,SV	SV,SE,P	See page 128	128
EVRSDDCTRL4	EVRC SD Control Register 4	0118 <sub>H</sub>	U,SV	SV,SE,P	See page 129	129
EVRSDDCTRL5	EVRC SD Control Register 5	011C <sub>H</sub>	U,SV	SV,SE,P	See page 130	130
EVRSDDCTRL6	EVRC SD Control Register 6	0120 <sub>H</sub>	U,SV	SV,SE,P	See page 133	133
EVRSDDCTRL7	EVRC SD Control Register 7	0124 <sub>H</sub>	U,SV	SV,SE,P	See page 134	134
EVRSDDCTRL8	EVRC SD Control Register 8	0128 <sub>H</sub>	U,SV	SV,SE,P	See page 136	136
EVRSDDCTRL9	EVRC SD Control Register 9	012C <sub>H</sub>	U,SV	SV,SE,P	See page 137	137
EVRSDDCTRL10	EVRC SD Control Register 10	0130 <sub>H</sub>	U,SV	SV,SE,P	See page 138	138
EVRSDDCOEFF0	EVRC SD Coefficient Register 0	0148 <sub>H</sub>	U,SV	SV,SE,P	See page 140	140
EVRSDDCOEFF1	EVRC SD Coefficient Register 1	014C <sub>H</sub>	U,SV	SV,SE,P	See page 141	141
EVRSDDCOEFF2	EVRC SD Coefficient Register 2	0150 <sub>H</sub>	U,SV	SV,SE,P	See page 143	143
EVRSDDCOEFF3	EVRC SD Coefficient Register 3	0154 <sub>H</sub>	U,SV	SV,SE,P	See page 145	145

## Power Management System for Low-End (PMSLE)

**Table 402 Register Overview - PMS (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
DTSSTAT	Die Temperature Sensor Status Register	01C0 <sub>H</sub>	U,SV	BE	See page <a href="#">146</a>	<a href="#">146</a>
DTSLIM	Die Temperature Sensor Limit Register	01C8 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">147</a>	<a href="#">147</a>
OTSS	OCDS Trigger Set Select Register	01E0 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">170</a>	<a href="#">170</a>
OTSC0	OCDS Trigger Set Control 0 Register	01E4 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">170</a>	<a href="#">170</a>
OTSC1	OCDS Trigger Set Control 1 Register	01E8 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">172</a>	<a href="#">172</a>
ACCENO	Access Enable Register 0	01F8 <sub>H</sub>	U,SV	SV,SE,32	Application Reset	<a href="#">173</a>
ACCEN1	Access Enable Register 1	01FC <sub>H</sub>	U,SV	SV,SE,32	Application Reset	<a href="#">174</a>

### 12.3.2.1 Safety Flip-Flops

Safety flip-flops are special flip-flops that implement a hardware mechanism capable to detect single event effects that may lead to single event upsets (bit flip). The configuration and control registers that are implemented with safety flip-flops are:

- [EVRRSTCON](#)
- [EVRTRIM](#)
- [EVRMONCTRL](#)
- [EVRMONFILT](#)
- [EVRUVMON](#)
- [EVROVMON](#)
- [EVRUVMON2](#)
- [EVROVMON2](#)
- [HSMUVMON](#)
- [HSMOVMON](#)
- [EVROSCCTRL](#)
- [EVRSDCTRL0](#)
- [EVRSDCTRL1](#)
- [EVRSDCTRL2](#)
- [EVRSDCTRL3](#)
- [EVRSDCTRL4](#)
- [EVRSDCTRL5](#)
- [EVRSDCTRL6](#)
- [EVRSDCTRL7](#)
- [EVRSDCTRL8](#)

## Power Management System for Low-End (PMSLE)

- **EVRSDCTRL9**
- **EVRSDCTRL10**
- **EVRSDCOEFF0**
- **EVRSDCOEFF1**
- **EVRSDCOEFF2**
- **EVRSDCOEFF3**
- **PMSWCR0**
- **PMSWCR5**

### 12.3.2.2 Power Supply Generation and Monitoring Control Registers

This section describes the kernel registers of the PMS module. Most of PMS kernel register names described in this section will be referenced in other parts of the Target Specification by the module name prefix “PMS\_”. All PMS registers are placed in the VDDPD Pre-Regulator domain. After a cold PORST, these registers may return the default isolation value or the updated value by the Firmware. Otherwise, a read will provide the value of the most recent write operation. In PMS subsystem some registers are reset with cold PORST which encompasses predominantly registers with EVR power generation and primary and secondary monitoring functions. This ensures that certain registers are not erroneously updated and the system does not end up in permanent reset situation. The registers covering standby and infrastructure functions however are reset with the EVR LVD (Low Voltage Detector Reset) master reset.

Tricore atomic instructions (LDMST, ST.T, SWAP.W, SWAPMASK.W, CMPSWAP.W) only write back bits that are changing their level. This leads to the fact that bits that are already set cannot be written with a 1 when using RMW instructions. No problem exists when using direct write instructions (e.g. ST.W). This affects the status bits in register DTSIM.LLU and UOF bits which are cleared by writing 1s.

#### Identification Register

ID	(0008 <sub>H</sub> )																Application Reset Value: 00E8 C001 <sub>H</sub>								
Identification Register	MODNUMBER																								
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	r								
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	r	MODTYPE		MODREV					

Field	Bits	Type	Description
<b>MODREV</b>	7:0	r	<b>Module Revision Number</b> This bit field indicates the revision number of the PMS module.
<b>MODTYPE</b>	15:8	r	<b>Module Type</b> This bit field is fixed coded as C0 <sub>H</sub> . It defines a 32-bit module.

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>MODNUMBER</b>	31:16	r	<b>Module Number</b> This bit field defines the module identification number. The identification number for the PMS is $00E8_H$ .

### EVR Status Register

The status registers EVRSTAT, EVRADCSTAT, EVRMONSTAT1, EVRMONSTAT2 and EVRSDSTAT0 are updated during Start-up and after every EVRx closed loop cycle with the actual status and therefore the read value may differ from the reset value. The over-voltage and under-voltage event flag signals are reported to SMU.EMM unit. An alarm for the upper and lower bound is supported in the SMU.EMM unit.

#### EVRSTAT

**EVR Status Register** (002C<sub>H</sub>) Reset Value: [Table 403](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	UVDD M	UVSB	UVPRE	OVDD M	OVSB	OVPR E	EVRCMOD	SDVO K	SWDL VL	EVR33 SHHV	EVR33 SHLV	EVRCSS HHV	EVRCSS HLV		
r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSTS WD	RST33	RSTC		0	EVR33 VOK	SYNCL CK	UVSW D	UV33	UVC	OVSW D	OV33	EVR33 OVC	OVC	EVRC	
rh	rh	rh	r		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
<b>EVRC</b>	0	rh	<b>EVRC status</b> This bit is set if the internal EVRC regulator is currently active. EVRC is activated if HWCFG[2] pin level is latched high during start-up phase. $0_B$ EVRC is inactive. $1_B$ EVRC is active.
<b>OVC</b>	1	rh	<b>VDD Over-voltage event flag</b> This bit is set if VDD secondary voltage monitor recognizes a over-voltage event. An alarm is raised to the SMU and the status bit remains set until violation disappears. $0_B$ No Over-voltage condition or event active. $1_B$ VDD Over-voltage condition event indication as configured in EVROVMON / EVRMONCTRL register.
<b>EVR33</b>	2	rh	<b>EVR33 status</b> This bit is set if the internal EVR33 LDO regulator is active. EVR33 is activated if HWCFG[1] pin level is latched high during start-up phase. $0_B$ EVR33 is inactive. $1_B$ EVR33 is active.

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
OV33	3	rh	<p><b>VDDP3 Over-voltage event flag</b></p> <p>This bit is set if VDDP3 secondary voltage monitor recognizes a over-voltage event. An alarm is raised to the SMU and the status bit remains set until violation disappears.</p> <p>0<sub>B</sub> No over-voltage condition or event active. 1<sub>B</sub> VDDP3 Over-voltage event indication as configured in EVROVMON / EVRMONCTRL register.</p>
OVSWD	4	rh	<p><b>VEXT Over-voltage event flag</b></p> <p>This bit is set if VEXT secondary voltage monitor recognizes an over-voltage event. An alarm is raised to the SMU and the status bit remains set until violation disappears.</p> <p>0<sub>B</sub> No over-voltage condition or event active. 1<sub>B</sub> VEXT Over-voltage event indication as configured in EVROVMON / EVRMONCTRL register.</p>
UVC	5	rh	<p><b>VDD Under-voltage event flag</b></p> <p>This bit is set if VDD secondary voltage monitor recognizes a under-voltage event. An alarm is raised to the SMU and the status bit remains set until violation disappears.</p> <p>0<sub>B</sub> No under-voltage condition or event active. 1<sub>B</sub> VDD Under-voltage event indication as configured in EVRUVMON / EVRMONCTRL register.</p>
UV33	6	rh	<p><b>VDDP3 Under-voltage event flag</b></p> <p>This bit is set if VDDP3 secondary voltage monitor recognizes a under-voltage event. An alarm is raised to the SMU and the status bit remains set until violation disappears.</p> <p>0<sub>B</sub> No under-voltage condition or event active. 1<sub>B</sub> VDDP3 Under-voltage event indication as configured in EVRUVMON / EVRMONCTRL register.</p>
UVSWD	7	rh	<p><b>VEXT Under-voltage event flag</b></p> <p>This bit is set if VEXT secondary voltage monitor recognizes an under-voltage event. An alarm is raised to the SMU and the status bit remains set until violation disappears.</p> <p>0<sub>B</sub> No under-voltage condition or event active. 1<sub>B</sub> VEXT Under-voltage event indication as configured in EVRUVMON / EVRMONCTRL register.</p>

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>SYNCLCK</b>	8	rh	<p><b>EVRC Synchronization Input Locked status(<code>sd_sync_in_locked_o</code>)</b></p> <p>This bitfield indicates the current synchronization status of EVRC SMPS regulator to external DCDCSYNCI input signal. When the EVRC switching frequency/ edge is locked to the synchronization input, the SYNCLCK bit is set to HIGH indicating the locked state. When the synchronization is lost owing to frequency deviations beyond MAXDEV or the feature is disabled via SYNCEN, the SYNCLCK bit is set to LOW.</p> <p>This EVRC Synchronization status is indicated in EVRSDSTAT0.SYNCLCK status bits.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> EVRC regulator runs on internal configured switching frequency and is not currently synchronized to external DCDCSYNCI input signal.</li> <li>1<sub>B</sub> EVRC regulator switching frequency is currently synchronized to external DCDCSYNCI input signal.</li> </ul>
<b>EVR33VOK</b>	9	rh	<p><b>EVR33 Regulator Voltage OK status</b></p> <p>This bit is set after the soft ramp-up time of the EVR33 voltage OK ramp detector has elapsed and is not based on the measured VDDP3 voltage at the end of ramp-phase..</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> EVR33 ramp-up time has not elapsed.</li> <li>1<sub>B</sub> EVR33 ramp-up time has elapsed.</li> </ul>
<b>RSTC</b>	13	rh	<p><b>EVRC Reset Trigger</b></p> <ul style="list-style-type: none"> <li>0<sub>B</sub> No cold reset trigger signal is active after spike filter and core VDD voltage output is above the selected reset trim value.</li> <li>1<sub>B</sub> A cold reset trigger signal is active after spike filter and core VDD voltage output is below the selected reset trim value.</li> </ul>
<b>RST33</b>	14	rh	<p><b>EVR33 Reset Trigger</b></p> <ul style="list-style-type: none"> <li>0<sub>B</sub> No cold reset trigger signal is active after spike filter and 3.3 V VDDP3 voltage output is above the selected reset trim value.</li> <li>1<sub>B</sub> A cold reset trigger signal is active after spike filter and 3.3 V VDDP3 voltage output is below the selected reset trim value.</li> </ul>
<b>RSTSVD</b>	15	rh	<p><b>EVR SWD Reset Trigger</b></p> <ul style="list-style-type: none"> <li>0<sub>B</sub> No cold reset trigger signal is active after spike filter and VEXT voltage input is above the selected reset trim value.</li> <li>1<sub>B</sub> A cold reset trigger signal is active after spike filter and VEXT voltage input is below the selected reset trim value.</li> </ul>
<b>EVRCSHLV</b>	16	rh	<p><b>Short to ground</b></p> <p>This bit is set if a short condition to ground has been detected. The measured EVRC output is below the operational supply range and the upper controller limits are reached. The feature is supported only during closed loop operation or EVRCMOD = 00b.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> No short to ground detected on VDD rail.</li> <li>1<sub>B</sub> Short to ground detected on VDD rail.</li> </ul>

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>EVRCSHHV</b>	17	rh	<p><b>Short to supply</b></p> <p>This bit is set if a short condition to supply has been detected. The measured EVRC output exceeds the allowed supply range and the lower controller limits are reached. The feature is supported only during closed loop operation or EVRCMOD = 00b.</p> <p><math>0_B</math> No short to supply detected on VDD rail.  <math>1_B</math> Short to supply detected on VDD rail.</p>
<b>EVR33SHLV</b>	18	rh	<p><b>Short to ground</b></p> <p>This bit is set if a short condition to ground has been detected. The measured EVR33 output is below the operational supply range and the lower gate drive threshold voltage driving P ch. MOSFET is reached.</p> <p><math>0_B</math> No short to ground detected on VDDP3 rail.  <math>1_B</math> Short to ground detected on VDDP3 rail.</p>
<b>EVR33SHHV</b>	19	rh	<p><b>Short to supply</b></p> <p>This bit is set if a short condition to supply has been detected. The measured EVR33 output exceeds the allowed supply range and the upper gate drive threshold voltage driving P ch. MOSFET is reached.</p> <p><math>0_B</math> No short to supply detected on VDDP3 rail.  <math>1_B</math> Short to supply detected on VDDP3 rail.</p>
<b>SWDLVL</b>	20	rh	<p><b>VEXT External Supply Level Status</b></p> <p>This bit indicates that the VEXT voltage has dropped below ~4 V to indicate EVRC parameter switch to differentiate 5V or 3.3V external supply. A hysteresis of ~120 mV is implemented on this detector.</p> <p><math>0_B</math> VEXT external supply is above the threshold.  <math>1_B</math> VEXT external supply is below the threshold.</p>
<b>SDVOK</b>	21	rh	<p><b>EVRC Regulator Voltage OK status</b></p> <p>This bit is set by the EVRC voltage OK detector to indicate that the new regulator output value has been reached. This bit is reset incase EVRTRIM, SDVOUTSEL or SDVOUTTRIM values are adapted to scale core voltage and is set when the new output setpoint is reached. This bit is also reset incase droop compensation is requested before a load jump event. A time out period of x us shall be waited when polling SDVOK bit.</p> <p><math>0_B</math> EVRC regulator setpoint voltage has not been reached.  <math>1_B</math> EVRC regulator setpoint voltage is reached and VDD voltage is ok.</p>
<b>EVRCMOD</b>	23:22	rh	<p><b>EVRC Mode</b></p> <p>EVRC Operation Mode. This bit-field indicates the current operation mode of the SC-DCDC: bypass switches disabled, bypass switches enabled.</p> <p><math>00_B</math> SMPS Normal SC Mode: The bypass switches are not active.  <math>01_B</math> SMPS operation with bypass switches activated. High current consumption expected. An interrupt is generated to the application SW if enabled in the PMSIEN register.  <math>10_B</math> Reserved.  <math>11_B</math> Reserved.</p>

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>OVPRE</b>	24	rh	<p><b>Pre Regulator VDDPD Over-voltage event flag</b></p> <p>This bit is set if VDDPD supply secondary voltage monitor recognizes an over-voltage event. An alarm is raised to the SMU.</p> <p>0<sub>B</sub> No over-voltage condition happened. 1<sub>B</sub> VDDPD Over-voltage event indication as configured in EVROVMON2 register.</p>
<b>OVSB</b>	25	rh	<p><b>Standby Supply or VEVRSB Over-voltage event flag</b></p> <p>This bit is set if VEVRSB supply secondary voltage monitor recognizes an over-voltage event. An alarm is raised to the SMU.</p> <p>0<sub>B</sub> No over-voltage condition happened. 1<sub>B</sub> VEVRSB Over-voltage event indication as configured in EVROVMON2 register.</p>
<b>OVDDM</b>	26	rh	<p><b>ADC VDDM Supply Over-voltage event flag</b></p> <p>This bit is set if VDDM ADC supply secondary voltage monitor recognizes an over-voltage event. An alarm is raised to the SMU.</p> <p>0<sub>B</sub> No over-voltage condition happened. 1<sub>B</sub> VDDM Over-voltage event indication as configured in EVROVMON2 register.</p>
<b>UVPRE</b>	27	rh	<p><b>Pre Regulator VDDPD Under-voltage event flag</b></p> <p>This bit is set if VDDPD supply secondary voltage monitor recognizes an under-voltage event. An alarm is raised to the SMU.</p> <p>0<sub>B</sub> No under-voltage condition happened. 1<sub>B</sub> VDDPD Under-voltage event indication as configured in EVRUVMON2 register.</p>
<b>UVSB</b>	28	rh	<p><b>Standby Supply or VEVRSB Under-voltage event flag</b></p> <p>This bit is set if VEVRSB supply secondary voltage monitor recognizes an under-voltage event. An alarm is raised to the SMU.</p> <p>0<sub>B</sub> No under-voltage condition happened. 1<sub>B</sub> VEVRSB Under-voltage event indication as configured in EVRUVMON2 register.</p>
<b>UVDDM</b>	29	rh	<p><b>ADC VDDM Supply Under-voltage event flag</b></p> <p>This bit is set if VDDM ADC supply secondary voltage monitor recognizes an under-voltage event. An alarm is raised to the SMU.</p> <p>0<sub>B</sub> No under-voltage condition happened. 1<sub>B</sub> VDDM Under-voltage event indication as configured in EVRUVMON2 register.</p>
<b>0</b>	12:10, 31:30	r	<p><b>Reserved</b></p> <p>Read as 0.</p>

**Table 403 Reset Values of EVRSTAT**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

## Power Management System for Low-End (PMSLE)

### EVR Primary ADC Status Register

#### EVRADCSTAT

#### EVR Primary ADC Status Register

(0034<sub>H</sub>)

LVD Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	UVSW D	UV33	UVC	OVSW D	OV33	OVC									ADCSWDV
r	rh	rh	rh	rh	rh	rh									rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															ADCCV
															rh

Field	Bits	Type	Description
<b>ADCCV</b>	7:0	rh	<b>ADC VDD Core Voltage Conversion Result</b> This bit field contains the last filtered conversion result of the ADC measurement of the VDD / EVRC supply by the Primary Monitor. VIN = [0.7125 + (ADCCV * LSB)] V LSB = 5 mV Eg. 1.25 V = 6C
<b>ADC33V</b>	15:8	rh	<b>ADC VDDP3 Voltage Conversion Result</b> This bit field contains the last filtered conversion result of the ADC measurement of the VDDP3 / EVR33 supply by the Primary Monitor. VIN = [0.9375 + (ADC33V * LSB)] V LSB = 15 mV Eg. 3.3 V = 9E
<b>ADCSWDV</b>	23:16	rh	<b>ADC VEXT Supply Conversion Result</b> This bit field contains the last filtered conversion result of the ADC measurement of the external VEXT (3.3V / 5V) supply by the Primary Monitor. VIN = [1.050 + (ADCSWDV * LSB)] V LSB = 20 mV Eg. 5 V = C6
<b>OVC</b>	24	rh	<b>EVRC Regulator or VDD Over-voltage event flag</b> This bit is set if VDD primary voltage monitor recognizes a over-voltage event. An alarm is raised to the HSM and SMU. 0 <sub>B</sub> No over-voltage condition happened. 1 <sub>B</sub> VDD Over-voltage event indication as configured in HSMOVMON register.
<b>OV33</b>	25	rh	<b>EVR33 Regulator or VDDP3 Over-voltage event flag</b> This bit is set if VDDP3 primary voltage monitor recognizes a over-voltage event. An alarm is raised to the HSM and SMU. 0 <sub>B</sub> No over-voltage condition happened. 1 <sub>B</sub> VDDP3 Over-voltage event indication as configured in HSMOVMON register.

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
OVSWD	26	rh	<b>Supply Watchdog (SWD) or VEXT Over-voltage event flag</b> This bit is set if VEXT primary voltage monitor recognizes an over-voltage event. An alarm is raised to the HSM and SMU. 0 <sub>B</sub> No over-voltage condition happened. 1 <sub>B</sub> VEXT Over-voltage event indication as configured in HSMOVMON register.
UVC	27	rh	<b>EVRC Regulator or VDD Under-voltage event flag</b> This bit is set if VDD primary voltage monitor recognizes a under-voltage event. An alarm is raised to the HSM and SMU. 0 <sub>B</sub> No under-voltage condition happened. 1 <sub>B</sub> VDD Under-voltage event indication as configured in HSMUVMON register.
UV33	28	rh	<b>EVR33 Regulator or VDDP3 Under-voltage event flag</b> This bit is set if VDDP3 primary voltage monitor recognizes a under-voltage event. An alarm is raised to the HSM and SMU. 0 <sub>B</sub> No under-voltage condition happened. 1 <sub>B</sub> VDDP3 Under-voltage event indication as configured in HSMUVMON register.
UVSWD	29	rh	<b>Supply Watchdog (SWD) or VEXT Under-voltage event flag</b> This bit is set if VEXT primary voltage monitor recognizes an under-voltage event. An alarm is raised to the HSM and SMU. 0 <sub>B</sub> No under-voltage condition happened. 1 <sub>B</sub> VEXT Under-voltage event indication as configured in HSMUVMON register.
0	31:30	r	<b>Reserved</b> Read as 0.

**EVR Reset Control Register**

This register allows the activation/deactivation of the primary monitor under-voltage resets for the external supply and the generated EVR33 and EVRC voltages. The respective reset threshold trim values are also configured in this register

**EVRRSTCON**
**EVR Reset Control Register**
**(003C<sub>H</sub>)**
**Reset Value: Table 405**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SLCK	<b>BPRST SWDO FF</b>	<b>RSTS WDOF F</b>	<b>BPRST 33OFF</b>	<b>RST33 OFF</b>	<b>BPRST COFF</b>	<b>RSTC OFF</b>								<b>RSTSVDTRIM</b>
r	rw	w	rw	w	rw	w	rw								rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>RST33TRIM</b>								<b>RSTCTRIM</b>							
rw								rw							

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>RSTCTRIM</b>	7:0	rw	<p><b>VDD Supply Reset Trim Value</b></p> <p>This bit field selects the hard reset generation level of VDD supply rail.</p> <p>This bit field is trimmed by Firmware.</p> <p><math>RSTCTRIM = [(VDDx - 712.5 \text{ mV}) / \text{LSB}]</math></p> <p><math>VDDPRIUV = 712.5 \text{ mV} + \text{LSB} * RSTCTRIM \text{ (signed value)}</math></p> <p>LSB = 5 mV</p>
<b>RST33TRIM</b>	15:8	rw	<p><b>VDDP3 Supply Reset Trim Value</b></p> <p>This bit field selects the hard reset generation level of VDDP3 supply rail.</p> <p>This bit field is trimmed by Firmware.</p> <p><math>RST33TRIM = [(VDDx - 937.5 \text{ mV}) / \text{LSB}]</math></p> <p><math>VDDP3PRIUV = 937.5 \text{ mV} + \text{LSB} * RST33TRIM + \text{LSB} * RST33PTRIM \text{ (signed value)}</math></p> <p>LSB = 15 mV</p>
<b>RSTSWDTRIM</b>	23:16	rw	<p><b>VEXT Supply Reset Trim Value</b></p> <p>This bit field selects the hard reset generation level of the external VEXT supply rail. This bitfield is trimmed by Firmware.</p> <p><math>RSTSWDTRIM = [(VDDx - 1050 \text{ mV}) / \text{LSB}]</math></p> <p><math>VEXTPRIUV = 1050 \text{ mV} + \text{LSB} * RSTSWDTRIM</math></p> <p>LSB = 20 mV</p>
<b>RSTCOFF</b>	24	rw	<p><b>VDD Reset Enable</b></p> <p>This bit can only be changed if bit BPRSTCOFF is set in parallel. RSTCOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application.</p> <p><math>0_B</math> A reset trigger signal is generated and forwarded to the SCU by primary monitor depending on the selected reset trim value.</p> <p><math>1_B</math> No reset trigger signal is generated and forwarded to the SCU by primary monitor depending on the selected reset trim value.</p>
<b>BPRSTCOFF</b>	25	w	<p><b>Bit Protection RSTCOFF</b></p> <p>Setting this bit enables that bit RSTCOFF can be changed in this write operation. This bit is read as zero.</p>
<b>RST33OFF</b>	26	rw	<p><b>VDDP3 Reset Enable</b></p> <p>This bit can only be changed if bit BPRST33OFF is set in parallel. The VDDP3 reset is disabled by application to support voltage drop up to nominal 3.0 V during cranking. RST33OFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application.</p> <p><math>0_B</math> A reset trigger signal is generated and forwarded to the SCU by primary monitor depending on the selected reset trim value.</p> <p><math>1_B</math> No reset trigger signal is generated and forwarded to the SCU by primary monitor depending on the selected reset trim value.</p>
<b>BPRST33OFF</b>	27	w	<p><b>Bit Protection RST33OFF</b></p> <p>Setting this bit enables that bit RST33OFF can be changed in this write operation. This bit read also as zero.</p>

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>RSTSWD OFF</b>	28	rw	<p><b>VEXT Reset Enable</b></p> <p>This bit can only be changed if bit BPRSTSWD OFF is set in parallel. RSTSWD OFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application.</p> <p>0<sub>B</sub> A reset trigger signal is generated and forwarded to the SCU by primary monitor depending on the selected reset trim value. 1<sub>B</sub> No reset trigger signal is generated and forwarded to the SCU by primary monitor depending on the selected reset trim value.</p>
<b>BPRSTSWD OF F</b>	29	w	<p><b>Bit Protection RSTSWD OFF</b></p> <p>Setting this bit enables that bit RSTSWD OFF can be changed in this write operation.</p> <p>This bit is read as zero.</p>
<b>SLCK</b>	30	rw	<p><b>HSM Security Lock</b></p> <p>If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011<sub>B</sub>). A set operation performed by any other master or software is ignored and the bit is kept as cleared.</p> <p>0<sub>B</sub> No lock active 1<sub>B</sub> Lock is active</p>
<b>0</b>	31	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

Table 404 Access Mode Restrictions of **EVRRSTCON** sorted by descending priority

Mode Name	Access Mode		Description
<b>SLCK = 0 and write 1 to BPRSTSWD OFF</b>	rw	RSTSWD OFF	
<b>SLCK = 0 and write 1 to BPRST33OFF</b>	rw	RST33OFF	
<b>SLCK = 0 and write 1 to BPRSTCOFF</b>	rw	RSTCOFF	
<b>SLCK = 0</b>	rw	RST33TRIM, RSTCTRIM, RSTSVDTRIM	
<b>SLCK = 0</b>	w	BPRST33OFF, BPRSTCOFF, BPRSTSWD OFF	
(default)	r	RST33OFF, RST33TRIM, RSTCOFF, RSTCTRIM, RSTSWD OFF, RSTSVDTRIM, SLCK	
	rX	BPRST33OFF, BPRSTCOFF, BPRSTSWD OFF	

## Power Management System for Low-End (PMSLE)

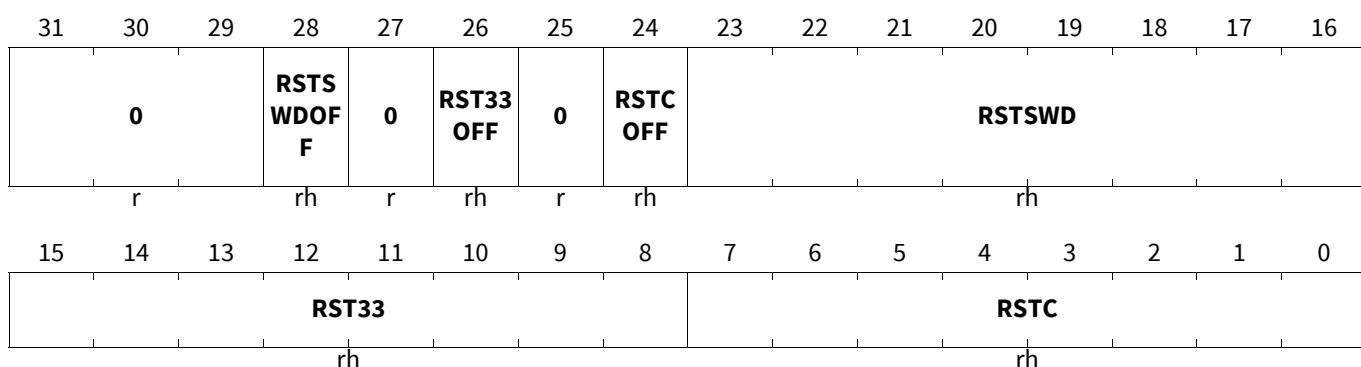
**Table 405 Reset Values of EVRRSTCON**

Reset Type	Reset Value	Note
LVD Reset	0059 7F4A <sub>H</sub>	
Cold PORST	0059 7F4A <sub>H</sub>	
After SSW execution	005C 834B <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVR Reset Status Register

#### EVRRSTSTAT

**EVR Reset Status Register** **(0044<sub>H</sub>)** **Reset Value: Table 406**



Field	Bits	Type	Description
<b>RSTC</b>	7:0	rh	<b>VDD Supply Reset Value Status</b> This bit field indicates the actual cold PORST reset trim setpoint for core voltage supply rail used by the Primary monitors. The value is updated via EVRRSTCON.RSTCTRIM register. RSTC = RSTCTRIM(signed value) RSTC range = 0 up to 255 $VDDPRIUV = 712.5 \text{ mV} + \text{LSB} * \text{RSTC}$ LSB = 5 mV
<b>RST33</b>	15:8	rh	<b>VDDP3 Supply Reset Value Status</b> This bit field indicates the actual cold PORST reset trim setpoint for 3.3 V supply rail used by the Primary monitors. The value is updated via EVRRSTCON.RST33TRIM register. RST33 = RST33TRIM + RST33PTRIM (signed value) RST33 range = 0 up to 255 $VDDP3PRIUV = 937.5 \text{ mV} + \text{LSB} * \text{RST33}$ LSB = 15 mV

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
RSTSWD	23:16	rh	<b>VEXT Supply Reset Value Status</b> This bit field indicates the actual cold PORST reset trim setpoint for 5 V supply rail used by the Primary monitors. The value is updated via EVRRSTCON.RSTSWDTRIM register. $RSTSWD = RSTSWDTRIM$ (signed value) RSTSWD range = 0 up to 255 $VEXTPRIUV = 1050 \text{ mV} + \text{LSB} * RSTSWD$ LSB = 20 mV
RSTCOFF	24	rh	<b>EVRC Reset Enable Status</b> The value is updated via EVRRSTCON.RSTCOFF register bit. 0 <sub>B</sub> A cold PORST is triggered incase of VDD primary under-voltage event 1 <sub>B</sub> No cold PORST is generated incase of a primary under-voltage event.
RST33OFF	26	rh	<b>EVR33 Reset Enable Status</b> The value is updated via EVRRSTCON.RST33OFF register bit. 0 <sub>B</sub> A cold PORST is triggered incase of VDDP3 primary under-voltage event 1 <sub>B</sub> No cold PORST is generated incase of a primary under-voltage event.
RSTSWDOFF	28	rh	<b>EVR SWD Reset Enable</b> The value is updated via EVRRSTCON.RSTSWDOFF register bit. 0 <sub>B</sub> A cold PORST is triggered incase of VEXT primary under-voltage event 1 <sub>B</sub> No cold PORST is generated incase of a primary under-voltage event.
0	25, 27, 31:29	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 406 Reset Values of EVRRSTSTAT**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

### EVR Trim Control Register

EVRTRIM and EVRRSTCON register may be used to generate voltage stress conditions to subject the modules to voltages beyond normal operating ranges.

## Power Management System for Low-End (PMSLE)

## EVRTRIM

## EVR Trim Control Register

(004C<sub>H</sub>)Reset Value: [Table 408](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	<b>SLCK</b>							<b>SDVOUTTRIM</b>		<b>0</b>					<b>EVR33VOUTTRIM</b>
rh	rw			rw				r				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								<b>SDVOUTSEL</b>							<b>EVR33VOUTSEL</b>
								rw				rw			

Field	Bits	Type	Description
<b>EVR33VOUTS EL</b>	7:0	rw	<b>EVR33 Regulator Output Voltage Target Value</b> The VDDP3 output level of the EVR33 LDO regulator. The ramp-up completion to the new target value is indicated via EVRSTAT.EVR33VOK bit. The (EVR33VOUTSEL + EVR33VOUTTRIM) setpoint value shall be programmed between 0x24 and 0xDA for valid closed loop PID regulator function. $3.3\text{ V} - 9\text{E}_H - 158_D$ $\text{EVR33VOUTSEL} = [(\text{VDDP3} - 937.5\text{ mV}) / \text{LSB}]$ $\text{VDDP3} = 937.5\text{ mV} + \text{LSB} * \text{EVR33VOUTSEL}$ $\text{LSB} = 15\text{ mV}$
<b>SDVOUTSEL</b>	15:8	rw	<b>EVRC Regulator Output Voltage Target Value</b> The VDD output level of the Step down regulator. $1.25\text{ V} - 6\text{C} - 108_D$ $\text{SDVOUTSEL} = [(\text{VDD} - 712.5\text{ mV}) / \text{LSB}]$ ; $\text{VDD} = 712.5\text{ mV} + \text{LSB} * \text{SDVOUTSEL}$ ; $\text{LSB} = 5\text{ mV}$ . This register bitfield requires a parameter update via EVRSDCTRL0.UP for transfer to EVRC SMPS shadow register. The reaching of the new target value is indicated via EVRSTAT.SDVOKE bit.
<b>EVR33VOUTT RIM</b>	21:16	rw	<b>EVR33 Regulator Output Voltage Trim Value</b> The 6 bit ADC BIST trimming value offset added to the EVR33 output level value installed by firmware from the flash. $\text{VDDP3 Setpoint} = \text{EVR33VOUTSEL} + \text{EVR33VOUTTRIM}$ (signed value) $\text{EVR33VOUTTRIM RANGE} = -32 \text{ to } 31$ LSB $\text{LSB} = 15\text{ mV}$
<b>SDVOUTTRIM</b>	29:24	rw	<b>EVRC Regulator Output Voltage Trim Value(vtrim_trim_i)</b> The 6 bit ADC BIST trimming value offset added to the EVRC output level value installed by firmware from the flash. The reaching of the new setpoint is indicated via EVRSTAT.SDVOKE bits $\text{VDD Setpoint} = \text{SDVOUTSEL} + \text{SDVOUTTRIM}$ (signed value) $\text{SDVOUTTRIM RANGE} = -32 \text{ to } 31$ LSB $\text{LSB} = 5\text{ mV}$ This register bitfield requires a parameter update via EVRSDCTRL0.UP for transfer to SMPS shadow register.

# **Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
SLCK	30	rw	<p><b>HSM Security Lock</b></p> <p>If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011<sub>B</sub>). A set operation performed by any other master or software is ignored and the bit is kept as cleared.</p> <p>0<sub>B</sub> No lock active 1<sub>B</sub> Lock is active</p>
LCK	31	rh	<p><b>Lock Status</b></p> <p>This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <p>0<sub>B</sub> The register is unlocked and can be updated 1<sub>B</sub> The register is locked and cannot be updated</p>
0	23:22	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 407 Access Mode Restrictions of EVRTRIM sorted by descending priority**

Mode Name	Access Mode	Description
<b>SLCK</b> = 0 and <b>LCK</b> = 0	rw	EVR33VOUTSEL, EVR33VOUTTRIM, SDVOUTSEL, SDVOUTTRIM
(default)	r	EVR33VOUTSEL, EVR33VOUTTRIM, SDVOUTSEL, SDVOUTTRIM, SLCK

**Table 408** Reset Values of **EVRTRIM**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
After SSW execution	0000 6C9E <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## EVR Trim Status Register

EVRTRIMSTAT

## EVR Trim Status Register

(0050<sub>H</sub>)

**Reset Value:** Table 409

This timing diagram illustrates the relationship between two digital outputs: SDVOUTTRIM and EVR33VOUTTRIM. The top row shows the timing for SDVOUTTRIM, starting at bit 31 and ending at bit 16. The bottom row shows the timing for EVR33VOUTTRIM, starting at bit 15 and ending at bit 0. Both rows have a label 'r' below the first bit and 'rh' below the last bit.

Bit	SDVOUTTRIM	EVR33VOUTTRIM
31	0	
30		
29		
28		
27		
26		
25		
24		
23		0
22		
21		
20		
19		
18		
17		
16		

Bit	SDVOUTSEL	EVR33VOUTSEL
15		
14		
13		
12		
11		
10		
9		
8		
7		
6		
5		
4		
3		
2		
1		
0		

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>EVR33VOUTS EL</b>	7:0	rh	<b>EVR33 Regulator Output Voltage Target Value</b> This bitfield indicates EVR33 output target value as configured in EVTRIM.EVR33VOUTSEL.
<b>SDVOUTSEL</b>	15:8	rh	<b>EVRC Regulator Output Voltage Target Value</b> This bit field indicates the EVRC output level of the Step down regulator as configured in EVTRIM.SDVOUTSEL. (vosel_target_o)
<b>EVR33VOUTT RIM</b>	21:16	rh	<b>EVR33 Regulator Output Voltage Trim Value</b> This bit field indicates the 6 bit ADC BIST trimming value offset added to the EVR33 output level value installed by firmware from flash configuration sector if production trimming is required.
<b>SDVOUTTRIM</b>	29:24	rh	<b>EVRC Regulator Output Voltage Trim Value(vtrim_trim_o)</b> This bit field indicates the 5 bit ADC BIST trimming value offset added to the EVRC output level value installed by firmware from flash configuration sector as configured in EVTRIM.SDVOUTTRIM.
<b>0</b>	23:22, 31:30	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 409 Reset Values of EVRTRIMSTAT**

Reset Type	Reset Value	Note
LVD Reset	0000 6C9E <sub>H</sub>	
Cold PORST	0000 6C9E <sub>H</sub>	

### EVR Secondary ADC Status Register 1

#### EVRMONSTAT1

**EVR Secondary ADC Status Register 1 (0060<sub>H</sub>) Reset Value: Table 410**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	<b>ACTVCNT</b>				<b>ADCSWDV</b>										
r				rh					rh						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>ADC33V</b>								<b>ADCCV</b>							
				rh					rh						

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>ADCCV</b>	7:0	rh	<p><b>VDD Supply Secondary ADC Conversion Result</b></p> <p>This bit field contains the last conversion result of the ADC measurement of the VDD / EVRC supply by the Secondary Monitor. This bitfield is updated if secondary over- or under-voltage monitoring is activated via EVRMONCTRL.EVRCxxMOD.</p> <p>VIN = [LSB * (ADCx-1)] ; Ideal LSB = 5.7692 mV</p> <p>Full Range : 1465 mV</p> <p>E.g. 1.25 V = DA</p>
<b>ADC33V</b>	15:8	rh	<p><b>VDDP3 Supply Secondary ADC Conversion Result</b></p> <p>This bit field contains the last conversion result of the ADC measurement of the VDDP3 / EVR33 supply by the Secondary Monitor. This bitfield is updated if secondary over- or under-voltage monitoring is activated via EVRMONCTRL.EVR33xxMOD.</p> <p>VIN = [LSB * (ADCx-1)] ; Ideal LSB = 15.00 mV</p> <p>Full Range : 3810 mV</p> <p>E.g. 3.30 V = DD</p>
<b>ADCSWDV</b>	23:16	rh	<p><b>VEXT Supply Secondary ADC Conversion Result</b></p> <p>This bit field contains the last conversion result of the ADC measurement of the external VEXT (3.3V / 5V) supply by the Secondary Monitor. This bitfield is updated if secondary over- or under-voltage monitoring is activated via EVRMONCTRL.SWDxxMOD.</p> <p>VIN = [LSB * (ADCx-1)] ; LSB = 23.077 mV</p> <p>Full Range : 5861 mV</p> <p>E.g. 5.01 V = DA 3.3 V = 90</p>
<b>ACTVCNT</b>	29:24	rh	<p><b>Secondary Monitor Activity Counter</b></p> <p>This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations.</p> <p>The total number of conversions ConvTot = <math>\sum [ChX * ChXFIL]</math>.</p> <p>The counter is reset to 0 on a ConvTot overflow.</p>
<b>0</b>	31:30	r	<p><b>Reserved</b></p> <p>Read as 0.</p>

**Table 410 Reset Values of EVRMONSTAT1**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

## Power Management System for Low-End (PMSLE)

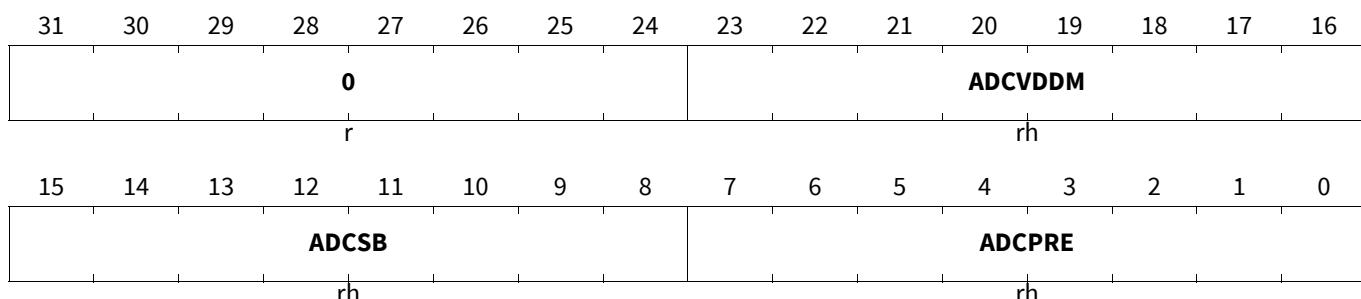
### EVR Secondary ADC Status Register 2

#### EVRMONSTAT2

#### EVR Secondary ADC Status Register 2

(0064<sub>H</sub>)

Reset Value: [Table 4.11](#)



Field	Bits	Type	Description
ADCPRE	7:0	rh	<b>VDDPD Supply Secondary ADC Conversion Result</b> This bit field contains the last conversion result of the ADC measurement of the VDDPD supply by the Secondary Monitor. This bitfield is updated if secondary over- or under-voltage monitoring is activated via EVRMONCTRL.PRExxMOD. $VIN = [\text{LSB} * (\text{ADCx}-1)]$ ; Ideal LSB = 5.7692 mV Full Range : 1465 mV E.g. 1.25 V = DA
ADCSB	15:8	rh	<b>VEVRSB Supply Secondary ADC Conversion Result</b> This bit field contains the last conversion result of the ADC measurement of the external VEVRSB (3.3V / 5V) standby supply by the Secondary Monitor. This bitfield is updated if secondary over- or under-voltage monitoring is activated via EVRMONCTRL.SBxxMOD. $VIN = [\text{LSB} * (\text{ADCx}-1)]$ ; Ideal LSB = 23.077 mV Full Range : 5861 mV E.g. 5.01 V = DA 3.0 V = 90
ADCVDDM	23:16	rh	<b>VDDM Supply Secondary ADC Conversion Result</b> This bit field contains the last conversion result of the ADC measurement of the VDDM ADC supply by the Secondary Monitor. This bitfield is updated if secondary over- or under-voltage monitoring is activated via EVRMONCTRL.VDDMxxMOD. $VIN = [\text{LSB} * (\text{ADCx}-1)]$ ; Ideal LSB = 23.077 mV Full Range : 5861 mV E.g. 5.01 V = DA <sub>D</sub> 3.0 V = 90 <sub>D</sub>
0	31:24	r	<b>Reserved</b> Read as 0.

## Power Management System for Low-End (PMSLE)

**Table 411 Reset Values of EVRMONSTAT2**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

### EVR Secondary Monitor Control Register

The default setting after reset is that over-voltage indication is notified via an SMU alarm when the over-voltage threshold is crossed in a lower to higher voltage transition. Overvoltage monitors use greater than equal compare if xOVMOD=01b or 11b and less than equal compare if xOVMOD=10b

The default setting after reset is that under-voltage indication is notified via an SMU alarm when the under-voltage threshold is crossed in a higher to lower voltage transition. Under voltage monitors use greater than equal compare if xUVMOD=01b and less than equal compare if xUVMOD=10b or 11b

It can be configured in EVRMONCTRL register to generate an interrupt when the over- under-voltage thresholds are crossed in either direction. This may be used to notify when the violation condition disappears with respect to secondary voltage monitoring. Interrupt is generated on low to high transition of the EVRSTAT monitoring bits incase of xOVMOD=01b or 10b and interrupt is generated on any transition incase of xOVMOD=11b.

### EVRMONCTRL

**EVR Secondary Monitor Control Register (0068<sub>H</sub>)** Reset Value: [Table 413](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>0</b>	<b>SLCK</b>			<b>0</b>				<b>SBUVMOD</b>	<b>SWDUVMOD</b>	<b>SBOVMOD</b>	<b>SWDOVMOD</b>				
r	rw			r				rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>VDDMUVMOD</b>	<b>EVR33UVMOD</b>	<b>VDDMOVMOD</b>	<b>EVR33OVMOD</b>	<b>PREUVMOD</b>	<b>EVRCUVMOD</b>	<b>PREOVMOD</b>	<b>EVRCOVMOD</b>								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

Field	Bits	Type	Description
<b>EVRCOVMOD</b>	1:0	rw	<b>VDD Over-voltage monitoring mode</b> Incase both EVRCOVMOD = 00 <sub>B</sub> & EVRCUVMOD = 00 <sub>B</sub> , then ADC conversion for the respective supply rail does not take place. 00 <sub>B</sub> Over-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted. 01 <sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used. 10 <sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used. 11 <sub>B</sub> An over-voltage event is triggered when the threshold is crossed in either direction. Greater than or equal compare is used.

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>PREOVMOD</b>	3:2	rw	<p><b>EVRPR or VDDPD Over-voltage monitoring mode</b>            Incase both PREOVMOD = 00<sub>B</sub> &amp; PREUVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Over-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in either direction. Greater than or equal compare is used.</li> </ul>
<b>EVRCUVMOD</b>	5:4	rw	<p><b>VDD Under-voltage monitoring mode</b>            Incase both EVRCOVMOD = 00<sub>B</sub> &amp; EVRCUVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Under-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in either direction. Less than or equal compare is used.</li> </ul>
<b>PREUVMOD</b>	7:6	rw	<p><b>EVRPR or VDDPD Under-voltage monitoring mode</b>            Incase both PREOVMOD = 00<sub>B</sub> &amp; PREUVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Under-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in either direction. Less than or equal compare is used.</li> </ul>

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>EVR330VMOD</b>	9:8	rw	<p><b>VDDP3 Supply Over-voltage monitoring mode</b></p> <p>Incase both EVR330VMOD = 00<sub>B</sub> &amp; EVR33UVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Over-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in either direction. Greater than or equal compare is used.</li> </ul>
<b>VDDMOVMOD</b>	11:10	rw	<p><b>VDDM ADC Supply Over-voltage monitoring mode</b></p> <p>Incase both VDDMOVMOD = 00<sub>B</sub> &amp; VDDMUVMOD = 00<sub>B</sub>, then ADC conversion for the VDDM supply rail continues to run as used for ADC function.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Over-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in either direction. Greater than or equal compare is used.</li> </ul>
<b>EVR33UVMOD</b>	13:12	rw	<p><b>VDDP3 Supply Under-voltage monitoring mode</b></p> <p>Incase both EVR330VMOD = 00<sub>B</sub> &amp; EVR33UVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Under-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in either direction. Less than or equal compare is used.</li> </ul>

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>VDDMUVMOD</b>	15:14	rw	<p><b>VDDM ADC Supply Under-voltage monitoring mode</b>            Incase both VDDMOVMOD = 00<sub>B</sub> &amp; VDDMUVMOD = 00<sub>B</sub>, then ADC conversion for the VDDM supply rail continues to run as used for ADC function.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Under-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in either direction. Less than or equal compare is used.</li> </ul>
<b>SWDOVMOD</b>	17:16	rw	<p><b>VEXT Over-voltage monitoring mode</b>            Incase both SWDOVMOD = 00<sub>B</sub> &amp; SWDUMVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Over-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in either direction. Greater than or equal compare is used.</li> </ul>
<b>SBOVMOD</b>	19:18	rw	<p><b>EVR Standby Supply or VEVRSB Over-voltage monitoring mode</b>            Incase both SBOVMOD = 00<sub>B</sub> &amp; SBUVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Over-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</li> <li>01<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</li> <li>10<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</li> <li>11<sub>B</sub> An over-voltage event is triggered when the threshold is crossed in either direction. Greater than or equal compare is used.</li> </ul>

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>SWDUVMOD</b>	21:20	rw	<p><b>VEXT Under-voltage monitoring mode</b></p> <p>Incase both SWDOVMOD = 00<sub>B</sub> &amp; SWDUVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <p>00<sub>B</sub> Under-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</p> <p>01<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</p> <p>10<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</p> <p>11<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in either direction. Less than or equal compare is used.</p>
<b>SBUVMOD</b>	23:22	rw	<p><b>EVR Standby Supply or VEVRSB Under-voltage monitoring mode</b></p> <p>Incase both SBOVMOD = 00<sub>B</sub> &amp; SBUVMOD = 00<sub>B</sub>, then ADC conversion for the respective supply rail does not take place.</p> <p>00<sub>B</sub> Under-voltage monitoring inactive. This results in a complete reset of the comparator unit, status bits and filter values and alarm is deasserted.</p> <p>01<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition. Greater than or equal compare is used.</p> <p>10<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition. Less than or equal compare is used.</p> <p>11<sub>B</sub> An under-voltage event is triggered when the threshold is crossed in either direction. Less than or equal compare is used.</p>
<b>SLCK</b>	30	rw	<p><b>HSM Security Lock</b></p> <p>If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011<sub>B</sub>). A set operation performed by any other master or software is ignored and the bit is kept as cleared.</p> <p>0<sub>B</sub> No lock active 1<sub>B</sub> Lock is active</p>
<b>0</b>	29:24, 31	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

## Power Management System for Low-End (PMSLE)

**Table 412 Access Mode Restrictions of EVRMONCTRL sorted by descending priority**

Mode Name	Access Mode		Description
SLCK = 0	rw	EVR33OVMOD, EVR33UVMOD, EVRCOVMOD, EVRCUVMOD, PREOVMOD, PREUVMOD, SBOVMOD, SBUVMOD, SWDOVMOD, SWDUMOD, VDDMOVMOD, VDDMUVMOD	
(default)	r	EVR33OVMOD, EVR33UVMOD, EVRCOVMOD, EVRCUVMOD, PREOVMOD, PREUVMOD, SBOVMOD, SBUVMOD, SLCK, SWDOVMOD, SWDUMOD, VDDMOVMOD, VDDMUVMOD	

**Table 413 Reset Values of EVRMONCTRL**

Reset Type	Reset Value	Note
LVD Reset	00A5 A5A5 <sub>H</sub>	
Cold PORST	00A5 A5A5 <sub>H</sub>	
After SSW execution	00A5 A5A5 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVR Secondary Monitor Filter Register

The assertion of alarm takes place when xFIL consecutive values are violating the threshold. Incase one of the values is not violating the threshold , the spike filter is reset. For renewed assertion of the alarm to take place, a repeated set of xFIL consecutive values violating the threshold are required.

#### EVRMONFILT

##### EVR Secondary Monitor Filter Register

(0070<sub>H</sub>)

Reset Value: [Table 415](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SLCK	CLRFIL L		0					SBFIL				SWDFIL		
r	rw	rw		r					rw				rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VDDMFIL				EVR33FIL				PREFIL				EVRCFIL			
rw				rw				rw				rw			

Field	Bits	Type	Description
EVRCFIL	3:0	rw	<b>VDD Secondary ADC Supply Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
PREFIL	7:4	rw	<b>VDDPD Secondary ADC Supply Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
EVR33FIL	11:8	rw	<b>VDDP3 Secondary ADC Supply Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
VDDMFIL	15:12	rw	<b>VDDM Secondary ADC Supply Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
SWDFIL	19:16	rw	<b>VEXT Secondary ADC Supply Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
SBFIL	23:20	rw	<b>VEVRSB Secondary ADC Supply Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
CLRFIL	29	rw	<b>Clear all Spike Filters</b> To avoid spurious alarms during change of configuration or start-up, CLRFIL shall be set followed by alarm reconfiguration followed by activation of filter logic by clearing CLRFIL register bit. $0_B$ No effect $1_B$ All spike filters configured in EVRMONFILT register are reset. The xFIL configuration value remains as configured and continue to be used for adc filtration.
SLCK	30	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011 <sub>B</sub> ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. $0_B$ No lock active $1_B$ Lock is active
0	28:24, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System for Low-End (PMSLE)

**Table 414 Access Mode Restrictions of EVRMONFILT sorted by descending priority**

Mode Name	Access Mode			Description
<b>SLCK = 0</b>	rw	CLRFIL, EVR33FIL, EVRCFIL, PREFIL, SBFIL, SWDFIL, VDDMFIL		
(default)	r	CLRFIL, EVR33FIL, EVRCFIL, PREFIL, SBFIL, SLCK, SWDFIL, VDDMFIL		

**Table 415 Reset Values of EVRMONFILT**

Reset Type	Reset Value	Note
LVD Reset	0000 0300 <sub>H</sub>	
Cold PORST	0000 0300 <sub>H</sub>	
After SSW execution	0001 0301 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## PMS Interrupt Enable Register

### PMSIEN

**PMS Interrupt Enable Register** (0074<sub>H</sub>) **Reset Value:** [Table 416](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SCRW DT	SCREC C	SCRRS T	SCRIN T	PINBW KP	PINAW KP	ESR1 WKP	ESR0 WKP	WUTW KP	0	SWDL VL	SYNCL CK	SDVO K	EVRC MOD	
r	rw	rw	rw	rw	rw	rw	rw	rw	rw	r	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				UVSB	OVSB	UVDD M	OVDD M	UVPRE	OVPR E	UVC	OVC	UV33	OV33	UVSW D	OVSW D

Field	Bits	Type	Description
<b>OVSWD</b>	0	rw	<b>OVSWD Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>UVSWD</b>	1	rw	<b>UVSWD Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.
<b>OV33</b>	2	rw	<b>OV33 Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. 0 <sub>B</sub> Interrupt is disabled. 1 <sub>B</sub> Interrupt is enabled.

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>UV33</b>	3	rw	<b>UV33 Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>OVC</b>	4	rw	<b>OVC Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>UVC</b>	5	rw	<b>UVC Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>OVPRE</b>	6	rw	<b>OVPRE Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>UVPRE</b>	7	rw	<b>UVPRE Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>OVDDM</b>	8	rw	<b>OVDDM Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>UVDDM</b>	9	rw	<b>UVDDM Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>OVSB</b>	10	rw	<b>OVSB Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>UVSB</b>	11	rw	<b>UVSB Interrupt enable</b> Interrupt triggered on event as configured in EVRMONCTRL register. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>EVRCMOD</b>	16	rw	<b>EVRCMOD Interrupt enable</b> Interrupt triggered on a state change of EVRSTAT.EVRCMOD[0] bitfield. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>SDVOK</b>	17	rw	<b>SDVOK Interrupt enable</b> Interrupt triggered on EVRSTAT.SDVOK rising edge event. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>SYNCLCK</b>	18	rw	<b>SD SYNCLCK Interrupt enable</b> Interrupt triggered on a state change of EVRSTAT.SYNCLCK bitfield. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>SWDLVL</b>	19	rw	<b>SWDLVL Interrupt enable</b> Interrupt triggered on a state change of EVRSTAT.SWDLVL bitfield. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>WUTWKP</b>	21	rw	<b>WUTWKP Interrupt enable</b> Interrupt triggered on a WUTCNT underflow event. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>ESR0WKP</b>	22	rw	<b>ESR0WKP Interrupt enable</b> Interrupt triggered on a ESR0WKP event. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>ESR1WKP</b>	23	rw	<b>ESR1WKP Interrupt enable</b> Interrupt triggered on a ESR1WKP event. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>PINAWKP</b>	24	rw	<b>PINAWKP Interrupt enable</b> Interrupt triggered on a PINAWKP event. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>PINBWKP</b>	25	rw	<b>PINBWKP Interrupt enable</b> Interrupt triggered on a PINBWKP event. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>SCRINT</b>	26	rw	<b>SCRINT Interrupt enable</b> Interrupt triggered on a SCRINT event triggered by SCR to PMS to decode information in PMSWCR2.SCRINT register. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>SCRRST</b>	27	rw	<b>SCRRST Interrupt enable</b> Interrupt triggered by SCR to PMS on an internal SCR software reset. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>SCRECC</b>	28	rw	<b>SCRECC Interrupt enable</b> Interrupt triggered by SCR to PMS on an internal RAM double bit ECC error. $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>SCRWDT</b>	29	rw	<b>SCRWDT Interrupt enable</b> Interrupt triggered by SCR to PMS on an internal SCR watchdog timeout error.  $0_B$ Interrupt is disabled. $1_B$ Interrupt is enabled.
<b>0</b>	15:12, 20, 31:30	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 416 Reset Values of PMSIEN**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

### EVR Secondary Under-voltage Monitor Register

A programmable threshold with upper and lower voltage bounds can be defined in EVROVMON and EVRUVMON registers for monitoring EVRC and EVR33 regulator outputs. Gain and Offset corrected thresholds can be evaluated from datasheet VxxMON parameters

#### EVRUVMON

##### EVR Secondary Under-voltage Monitor Register (0078<sub>H</sub>)

Reset Value: [Table 418](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>0</b>	<b>SLCK</b>			<b>0</b>											<b>SWDUVVAL</b>
r	rw			r											rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EVRCUVVAL</b>										<b>EVRCUVVAL</b>					
rw										rw					

Field	Bits	Type	Description
<b>EVRCUVVAL</b>	7:0	rw	<b>VDD Supply Secondary Monitor Under-voltage threshold</b> This field defines the under-voltage monitoring threshold level of the EVRC regulator output or VDD supply. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 5.7692 mV
<b>EVRCUVVAL</b>	15:8	rw	<b>VDDP3 Supply Secondary Monitor Under-voltage threshold</b> This field defines the under-voltage monitoring threshold level of the EVR33 regulator output or VDDP3 supply. Ideal Threshold = [(VIN / LSB) + 1]. Ideal LSB = 15.00 mV

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>SWDUVVAL</b>	23:16	rw	<b>VEXT Supply Secondary Monitor Under-voltage threshold</b> This field defines the under-voltage threshold level of the external VEXT supply monitor. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 23.077 mV.
<b>SLCK</b>	30	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011 <sub>B</sub> ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. 0 <sub>B</sub> No lock active 1 <sub>B</sub> Lock is active
<b>0</b>	29:24, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 417 Access Mode Restrictions of EVRUVMON sorted by descending priority**

Mode Name	Access Mode		Description
<b>SLCK = 0</b>	rw	EVR33UVVAL, EVRCUVVAL, SWDUVVAL	
(default)	r	EVR33UVVAL, EVRCUVVAL, SLCK, SWDUVVAL	

**Table 418 Reset Values of EVRUVMON**

Reset Type	Reset Value	Note
LVD Reset	0075 A7B8 <sub>H</sub>	
Cold PORST	0075 A7B8 <sub>H</sub>	
After SSW execution	0075 A7B8 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

**EVR Secondary Over-voltage Monitor Register**
**EVROVMON**
**EVR Secondary Over-voltage Monitor Register (007C<sub>H</sub>)**
**Reset Value: Table 420**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>0</b>	<b>SLCK</b>			<b>0</b>											<b>SWDOVAL</b>
r	rw			r											rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EVROVVAL</b>								<b>EVRCOVVAL</b>							
rw								rw							

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>EVRCOVVAL</b>	7:0	rw	<b>VDD Supply Secondary Monitor Over-voltage threshold</b> This field defines the over-voltage monitoring threshold level of the EVRC regulator output or VDD supply. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 5.7692 mV
<b>EVR33OWVAL</b>	15:8	rw	<b>VDDP3 Supply Secondary Monitor Over-voltage threshold</b> This field defines the over-voltage monitoring threshold level of the EVR33 regulator output or VDDP3 supply. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 15.00 mV
<b>SWDOVVAL</b>	23:16	rw	<b>VEXT Supply Secondary Monitor Over-voltage threshold</b> This field defines the over-voltage threshold level of the external VEXT supply monitor. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 23.077 mV.
<b>SLCK</b>	30	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011 <sub>B</sub> ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. 0 <sub>B</sub> No lock active 1 <sub>B</sub> Lock is active
<b>0</b>	29:24, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 419 Access Mode Restrictions of EVROVMON sorted by descending priority**

Mode Name	Access Mode		Description
<b>SLCK = 0</b>	rw	EV33OWVAL, EVRCOVVAL, SWDOVVAL	
(default)	r	EV33OWVAL, EVRCOVVAL, SLCK, SWDOVVAL	

**Table 420 Reset Values of EVROVMON**

Reset Type	Reset Value	Note
LVD Reset	00FE FEEF <sub>H</sub>	
Cold PORST	00FE FEEF <sub>H</sub>	
After SSW execution	00FE FEEF <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## Power Management System for Low-End (PMSLE)

### EVR Secondary Under-voltage Monitor Register 2

#### EVRUVMON2

#### EVR Secondary Under-voltage Monitor Register 2(0080<sub>H</sub>)

Reset Value: [Table 422](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SLCK			VDDMLVLSEL							SBUVVAL				
r	rw			rw							rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				VDDMUUVVAL							PREUVVAL				
				rw							rw				

Field	Bits	Type	Description
PREUVVAL	7:0	rw	<b>VDDPD Supply Secondary Monitor Under-voltage threshold</b> This field defines the under-voltage monitoring threshold level of the VDDPD supply or EVRPR output. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 5.7692 mV
VDDMUUVVAL	15:8	rw	<b>VDDM Supply Secondary Monitor Under-voltage threshold</b> This field defines the under-voltage monitoring threshold level of the VDDM ADC supply. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 23.077 mV
SBUVVAL	23:16	rw	<b>VEVRSB Supply Secondary Monitor Under-voltage threshold</b> This field defines the under-voltage threshold level of the external VEVRSB (3.3V / 5V) standby supply monitor. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 23.077 mV.
VDDMLVLSEL	29:24	rw	<b>VDDM Level Select</b> This field defines the under-voltage monitoring threshold level required by EVADC / EDSADC modules to differentiate between 5 V or 3.3 V VDDM supply level to adjust analog behavior to the actual voltage level. The 6 MSB bits of the ADC result is compared against VDDMLVLSEL with 4 LSB hysteresis. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 92.308 mV
SLCK	30	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011 <sub>B</sub> ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. 0 <sub>B</sub> No lock active 1 <sub>B</sub> Lock is active

## **Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
0	31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 421 Access Mode Restrictions of EVRUVMON2 sorted by descending priority**

<b>Mode Name</b>	<b>Access Mode</b>		<b>Description</b>
<b>SLCK = 0</b>	rw	PREUVVAL, SBUVVAL, VDDMLVLSEL, VDDMUUVVAL	
(default)	r	PREUVVAL, SBUVVAL, SLCK, VDDMLVLSEL, VDDMUUVVAL	

**Table 422** Reset Values of **EVRUVMON2**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
LVD Reset	2A70 00BC <sub>H</sub>	
Cold PORST	2A70 00BC <sub>H</sub>	
After SSW execution	2A70 00BC <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## EVR Secondary Over-voltage Monitor Register 2

EVROVMON2

EVR Secondary Over-voltage Monitor Register 2(0084<sub>H</sub>)

**Reset Value:** [Table 424](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>0</b>	<b>SLCK</b>			<b>0</b>								<b>SBOVAL</b>			
r	rw			r								rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												<b>VDDMOVVAL</b>		<b>PREOVAL</b>	
												rw		rw	

Field	Bits	Type	Description
<b>PREOVVAL</b>	7:0	rw	<p><b>VDDPD Supply Secondary Monitor Over-voltage threshold</b></p> <p>This field defines the over-voltage monitoring threshold level of the VDDPD supply or EVRPR output.</p> <p>Ideal Threshold = [(VIN / LSB) + 1]</p> <p>Ideal LSB = 5.7692 mV</p>
<b>VDDMOVVAL</b>	15:8	rw	<p><b>VDDM Supply Secondary Monitor Over-voltage threshold</b></p> <p>This field defines the over-voltage monitoring threshold level of the VDDM ADC supply</p> <p>Ideal Threshold = [(VIN / LSB) + 1]</p> <p>Ideal LSB = 23.077 mV</p>

### Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
SBOVVAL	23:16	rw	<b>VEVRSB Supply Secondary Monitor Over-voltage threshold</b> This field defines the over-voltage threshold level of the external VEVRSB (3.3V / 5V) standby supply monitor. Ideal Threshold = [(VIN / LSB) + 1] Ideal LSB = 23.077 mV
SLCK	30	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011 <sub>B</sub> ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. 0 <sub>B</sub> No lock active 1 <sub>B</sub> Lock is active
0	29:24, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 423 Access Mode Restrictions of EVROVMON2 sorted by descending priority**

Mode Name	Access Mode		Description
SLCK = 0	rw	PREOVVAL, SBOVVAL, VDDMOVVAL	
(default)	r	PREOVVAL, SBOVVAL, SLCK, VDDMOVVAL	

**Table 424 Reset Values of EVROVMON2**

Reset Type	Reset Value	Note
LVD Reset	00FE FFEE <sub>H</sub>	
Cold PORST	00FE FFEE <sub>H</sub>	
After SSW execution	00FE FFEE <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVR Primary HSM Under-voltage Monitor Register

#### HSMUVMON

#### EVR Primary HSM Under-voltage Monitor Register(0088<sub>H</sub>)

Reset Value: [Table 426](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLCK	HSMFIL			SWDOFF	EVR33OFF	EVRCOFF	SWDUVVAL								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVR33UVVAL								EVRCUVVAL							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>EVRCUVVAL</b>	7:0	rw	<b>VDD Supply Primary Monitor Alarm Under-voltage threshold</b> This field defines the under-voltage monitoring threshold level of the EVRC regulator output or VDD supply. $EVRCUVVAL = [(VDDx - 712.5 \text{ mV}) / \text{LSB}]$ LSB = 5 mV
<b>EVR33UVVAL</b>	15:8	rw	<b>VDDP3 Supply Primary Monitor Alarm Under-voltage threshold</b> This field defines the under-voltage monitoring threshold level of the EVR33 regulator output or VDDP3 supply. $EVR33UVVAL = [(VDDx - 937.5 \text{ mV}) / \text{LSB}]$ LSB = 15 mV
<b>SWDUVVAL</b>	23:16	rw	<b>VEXT Supply Primary Monitor Alarm Under-voltage threshold</b> This field defines the under-voltage threshold level of the external VEXT supply monitor. $SWDUVVAL = [(VDDx - 1050 \text{ mV}) / \text{LSB}]$ LSB = 20 mV
<b>EVRCOFF</b>	24	rw	<b>VDD Primary Monitor UV Alarm Disable</b> $0_B$ A alarm trigger signal is generated and forwarded to the HSM by the EVRC block depending on the EVRCUVVAL configured value. $1_B$ No alarm trigger signal is generated and forwarded to the HSM by the EVRC block depending on the selected reset trim value.
<b>EVR33OFF</b>	25	rw	<b>VDDP3 Primary Monitor UV Alarm Disable</b> $0_B$ A alarm trigger signal is generated and forwarded to the HSM by the EVR33 block depending on the EVR33UVVAL configured value. $1_B$ No alarm trigger signal is generated and forwarded to the HSM by the EVR33 block depending on the selected reset trim value.
<b>SWDOFF</b>	26	rw	<b>VEXT Primary Monitor UV Alarm Disable</b> $0_B$ A alarm trigger signal is generated and forwarded to the HSM by the SWD block depending on the SWDUVVAL configured value. $1_B$ No alarm trigger signal is generated and forwarded to the HSM by the SWD block depending on the selected reset trim value.
<b>HSMFIL</b>	30:27	rw	<b>HSM Voltage Filter</b> $0_H$ Each conversion result is compared with threshold to generate alarm $F_H$ A spike filter of consecutive 16 ADC results are used to generate alarm to HSM.
<b>SLCK</b>	31	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master ( $TAG = 000011_B$ ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. $0_B$ No lock active $1_B$ Lock is active

## Power Management System for Low-End (PMSLE)

**Table 425 Access Mode Restrictions of HSMUVMON sorted by descending priority**

Mode Name	Access Mode			Description
SLCK = 0	rw	EVR33OFF, EVR33UVVAL, EVRCOFF, EVRCUVVAL, HSMFIL, SWDOFF, SWDUVVAL		
(default)	r	EVR33OFF, EVR33UVVAL, EVRCOFF, EVRCUVVAL, HSMFIL, SLCK, SWDOFF, SWDUVVAL		

**Table 426 Reset Values of HSMUVMON**

Reset Type	Reset Value	Note
LVD Reset	005C 824D <sub>H</sub>	
Cold PORST	005C 824D <sub>H</sub>	
After SSW execution	005C 824D <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVR Primary HSM Over-voltage Monitor Register

#### HSMOVMON

#### EVR Primary HSM Over-voltage Monitor Register(008C<sub>H</sub>)

Reset Value: [Table 428](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>SLCK</b>	<b>0</b>				<b>SWDOFF</b>	<b>EVR33OFF</b>	<b>EVRCOFF</b>	<b>SWDOVVAL</b>							
rw		r			rw	rw	rw								rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EVR33OVVAL</b>								<b>EVRCOVVAL</b>							
rw								rw							

Field	Bits	Type	Description
<b>EVRCOVVAL</b>	7:0	rw	<b>VDD Supply Primary Monitor Alarm Over-voltage threshold</b> This field defines the over-voltage monitoring threshold level of the EVRC regulator output or VDD supply. $EVRCOVVAL = [(VDDx - 712.5 \text{ mV}) / \text{LSB}]$ LSB = 5 mV
<b>EVR33OVVAL</b>	15:8	rw	<b>VDDP3 Supply Primary Monitor Alarm Over-voltage threshold</b> This field defines the over-voltage monitoring threshold level of the EVR33 regulator output or VDDP3 supply. $EVR33OVVAL = [(VDDx - 937.5 \text{ mV}) / \text{LSB}]$ LSB = 15 mV

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>SWDOVVAL</b>	23:16	rw	<b>VEXT Supply Primary Monitor Alarm Over-voltage threshold</b> This field defines the over-voltage threshold level of the external VEXT supply monitor. $SWDOVVAL = [(VDDx - 1050 \text{ mV}) / \text{LSB}]$ LSB = 20 mV
<b>EVRCOFF</b>	24	rw	<b>VDD Primary Monitor OV Alarm Disable</b> $0_B$ A alarm trigger signal is generated and forwarded to the HSM by the EVRC block depending on the EVRCOVVAL configured value. $1_B$ No alarm trigger signal is generated and forwarded to the HSM by the EVRC block depending on the selected reset trim value.
<b>EVR33OFF</b>	25	rw	<b>VDDP3 Primary Monitor OV Alarm Disable</b> $0_B$ A alarm trigger signal is generated and forwarded to the HSM by the EVR33 block depending on the EVR33OVVAL configured value. $1_B$ No alarm trigger signal is generated and forwarded to the HSM by the EVR33 block depending on the selected reset trim value.
<b>SWDOFF</b>	26	rw	<b>VEXT Primary Monitor OV Alarm Disable</b> $0_B$ A alarm trigger signal is generated and forwarded to the HSM by the SWD block depending on the SWDOVVAL configured value. $1_B$ No alarm trigger signal is generated and forwarded to the HSM by the SWD block depending on the selected reset trim value.
<b>SLCK</b>	31	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master ( $TAG = 000011_B$ ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. $0_B$ No lock active $1_B$ Lock is active
<b>0</b>	30:27	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 427 Access Mode Restrictions of [HSMOVMON](#) sorted by descending priority**

Mode Name	Access Mode		Description
<b>SLCK = 0</b>	rw	EVR33OFF, EVR33OVVAL, EVRCOFF, EVRCOVVAL, SWDOFF, SWDOVVAL	
(default)	r	EVR33OFF, EVR33OVVAL, EVRCOFF, EVRCOVVAL, SLCK, SWDOFF, SWDOVVAL	

## Power Management System for Low-End (PMSLE)

**Table 428 Reset Values of HSMOVMON**

Reset Type	Reset Value	Note
LVD Reset	00E1 B586 <sub>H</sub>	
Cold PORST	00E1 B586 <sub>H</sub>	
After SSW execution	00E1 B586 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVR33 Control Register

#### EVR33CON

**EVR33 Control Register** (0090<sub>H</sub>) Reset Value: [Table 430](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>0</b>	<b>SLCK</b>	<b>RES</b>	<b>RES</b>			<b>RES</b>						<b>SHVL33</b>			
r	rw	w	rw			rw						rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>RES</b>	<b>SHLVE N</b>	<b>SHHV EN</b>			<b>RES</b>							<b>SHVH33</b>			
rw	rw	rw			rw							rw			

Field	Bits	Type	Description
<b>SHVH33</b>	7:0	rw	<b>Short to Supply Voltage Threshold(x_i)</b> This field defines the upper threshold level VDDP3 supply. EVR33 short to supply alarm has the nominal values of SHVH33 = 4.5V and t33SHHV = 3ms. Do not change the reset value. $SHVH33 = [(VDDx - 937.5 \text{ mV}) / \text{ LSB}]$ LSB = 15 mV
<b>RES</b>	11:8, 15:14, 27:24, 28	rw	<b>Reserved</b> Must be written with original content.
<b>SHHVEN</b>	12	rw	<b>Short to High Detection Enable</b> $0_B$ Short to High Detection is disabled $1_B$ Short to High Detection is enabled
<b>SHLVEN</b>	13	rw	<b>Short to Low Detection Enable</b> $0_B$ Short to Low Detection is disabled $1_B$ Short to Low Detection is enabled
<b>SHVL33</b>	23:16	rw	<b>Short to Ground Voltage Threshold(x_i)</b> This field defines the lower threshold level VDDP3 supply. EVR33 short to ground alarm has the nominal values of SHVL33 = 1V and t33SHLV = 3ms. Do not change the reset value. $SHVL33 = [(VDDx - 937.5 \text{ mV}) / \text{ LSB}]$ LSB = 15 mV

### Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>RES</b>	29	w	<b>Reserved</b> This bit is read as zero, must be written with zero.
<b>SLCK</b>	30	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master ( $TAG = 000011_B$ ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. 0 <sub>B</sub> No lock active 1 <sub>B</sub> Lock is active
<b>0</b>	31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 429 Access Mode Restrictions of EVR33CON sorted by descending priority**

Mode Name	Access Mode		Description
<b>SLCK = 0</b>	w	, RES	
<b>SLCK = 0</b>	rw	, RES, SHHVEN, SHLVEN, SHVH33, SHVL33	
<b>SLCK = 0 and write 1 to BPEVR33OFF</b>	rw	, RES	
(default)	r	, RES, SHHVEN, SHLVEN, SHVH33, SHVL33, SLCK	
	rX		

**Table 430 Reset Values of EVR33CON**

Reset Type	Reset Value	Note
LVD Reset	0004 07ED <sub>H</sub>	
Cold PORST	0004 07ED <sub>H</sub>	
After SSW execution	0004 07ED <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## Power Management System for Low-End (PMSLE)

### EVR Oscillator Control Register

#### EVROSCCTRL

#### EVR Oscillator Control Register

(00A0<sub>H</sub>)

Reset Value: [Table 431](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>OSCT RIMEN</b>	0	<b>OSCTE MPOF FS</b>				0									<b>OSCFPTRIM</b>
rw	r	rw			r										rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				0											<b>OSCFTTRIM</b>
				r											rw

Field	Bits	Type	Description
<b>OSCFTTRIM</b>	5:0	rw	<b>Back-up Clock Fine Trim Value</b> This thermometer coded bit field contains information about the 100MHz OSC fine trimming. $f_{BACK\ ftrim} = [(OSCFTTRIM + (OSCFPTRIM)) * LSBFT] \text{ MHz}$ ; LSBFT = 110kHz Back-up Clock accuracy is documented in datasheet. It is recommended to wait 1 us after every fine trim step so that the clock source settles at the new frequency. $f_{BACK\ ftrim}$ value is saturated to range of 64. $00_H$ 0 MHz $1F_H$ 3.65 MHz $3F_H$ 7.3 MHz
<b>OSCFPTRIM</b>	21:16	rw	<b>OSC Fine Trim Signed Value</b> This bit field allows device individual trimming of the oscillator trim value during application. After updating the trim value, a waiting time of 1 us is required for the change to take effect.
<b>OSCTEMPOFF S</b>	29	rw	<b>Oscillator Temperature Offset Coefficient</b> This bitfield enables the centering function of the HPOSOC temperature coefficient to compensate for technology variations. $0_B$ Centering on. $1_B$ Centering off.
<b>OSCTRIMEN</b>	31	rw	<b>Dynamic Oscillator Trim Enable</b> Based on temperature, Oscillator can be trimmed. $0_B$ The Dynamic Oscillator Trim function is disabled/switched off. $1_B$ The Dynamic Oscillator Trim function is enabled.
0	15:6, 28:22, 30	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System for Low-End (PMSLE)

**Table 431 Reset Values of EVROSCCTRL**

Reset Type	Reset Value	Note
LVD Reset	0000 001F <sub>H</sub>	
After SSW execution	2000 001F <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

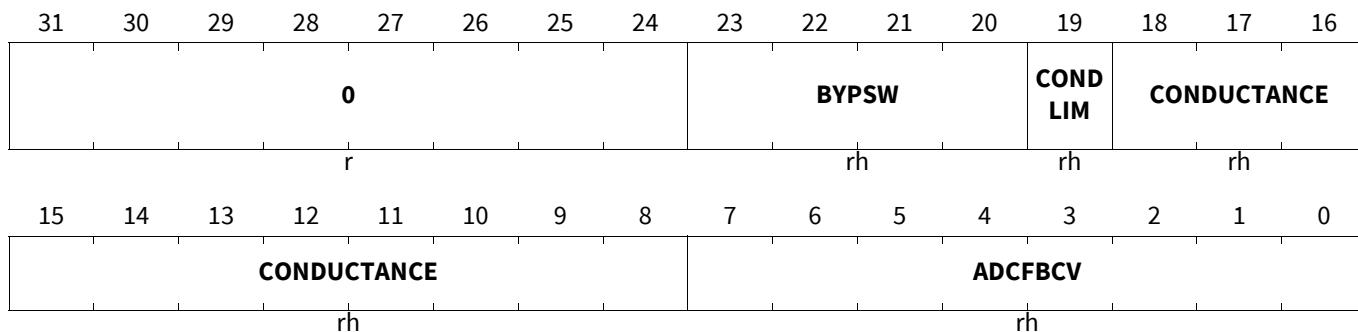
### EVR SD Status Register 0

#### EVRSDSTAT0

#### EVR SD Status Register 0

(00FC<sub>H</sub>)

Reset Value: [Table 432](#)



Field	Bits	Type	Description
ADCFBCV	7:0	rh	<b>Step Down Converter Core Voltage Feedback ADC Conversion Result, after the LPF (fbadc_data_lpf_o)</b> This bit field indicates the last ADC conversion result of the step down converter feedback ADC measuring VDD core voltage. VIN = [LSB * (ADCFBCV - EVRTRIM.SDVOUTTRIM) + 0.7125] V ; LSB = 5 mV E.g. 1.20 V - 62 - 98
CONDUCTANCE	18:8	rh	<b>Conductance value (conductance_o)</b> Conductance value calculated by the digital controller.
CONDLIM	19	rh	<b>Conductance limitation indication (cond_lim_on_o)</b> Indication if conductance is limited.
BYPSW	23:20	rh	<b>Number of closed bypass switches (bypass_sw_o)</b> Number of closed bypass switches.
0	31:24	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 432 Reset Values of EVRSDSTAT0**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

## Power Management System for Low-End (PMSLE)

### EVRC SD Control Register 0

#### EVRSCTRL0

#### EVRC SD Control Register 0

(0108<sub>H</sub>)

Reset Value: [Table 434](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	<b>UP</b>							<b>0</b>							
rh	rwh							r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					<b>0</b>							<b>SDFREQSPRD</b>			rw
					r										

Field	Bits	Type	Description
<b>SDFREQSPRD</b>	3:0	rw	<p><b>Frequency Spread Threshold(freq_spread_mode_i)</b>  This bit field defines the additional frequency spread to the nominal EVRC regulator switching frequency during operation. A random number of switching clock cycles are added to both TON and TOFF respectively.  where SDSWPRDNOM is the Nominal Switching period without spreading = TON + TOFF + 18</p> <ul style="list-style-type: none"> <li>0<sub>H</sub> no frequency spreading activated</li> <li>1<sub>H</sub> SDSWPRDNOM - (SDSWPRDNOM+2) switching period spread</li> <li>2<sub>H</sub> (SDSWPRDNOM-2) - (SDSWPRDNOM+2) switching period spread</li> <li>3<sub>H</sub> (SDSWPRDNOM-2) - (SDSWPRDNOM+4) switching period spread</li> <li>4<sub>H</sub> (SDSWPRDNOM-4) - (SDSWPRDNOM+4) switching period spread</li> <li>5<sub>H</sub> (SDSWPRDNOM-4) - (SDSWPRDNOM+6) switching period spread</li> <li>6<sub>H</sub> (SDSWPRDNOM-6) - (SDSWPRDNOM+6) switching period spread</li> <li>7<sub>H</sub> (SDSWPRDNOM-6) - (SDSWPRDNOM+8) switching period spread</li> <li>8<sub>H</sub> (SDSWPRDNOM-8) - (SDSWPRDNOM+8) switching period spread</li> <li>9<sub>H</sub> (SDSWPRDNOM-8) - (SDSWPRDNOM+10) switching period spread</li> <li>A<sub>H</sub> (SDSWPRDNOM-10) - (SDSWPRDNOM+10) switching period spread</li> </ul>
<b>UP</b>	30	rwh	<p><b>Update request for SMPS register values(param_update_i)</b>  This bitfield triggers the update of the current register values from PMSLE-FPI EVRC registers to the local SMPS module registers.  It shall be ensured that ALL EVRSCTRLx and EVRSDCOEFFx registers have correct and coherent values across the various registers before the update request is issued. Incase of singular register update, the other register values should match and be consistent. After a cold PORST, the UP bit is set as default reset value to ensure that the complete SMPS regulator parameter set is in its reset state. The parameter update via UP bit is not allowed in start-up and low power mode.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> No action is undertaken.</li> <li>1<sub>B</sub> A new complete EVRC parameter set is transferred to the SMPS module. All EVRSCTRLx and EVRSDCOEFFx register contents are transferred.</li> </ul>

**Power Management System for Low-End (PMSLE)**

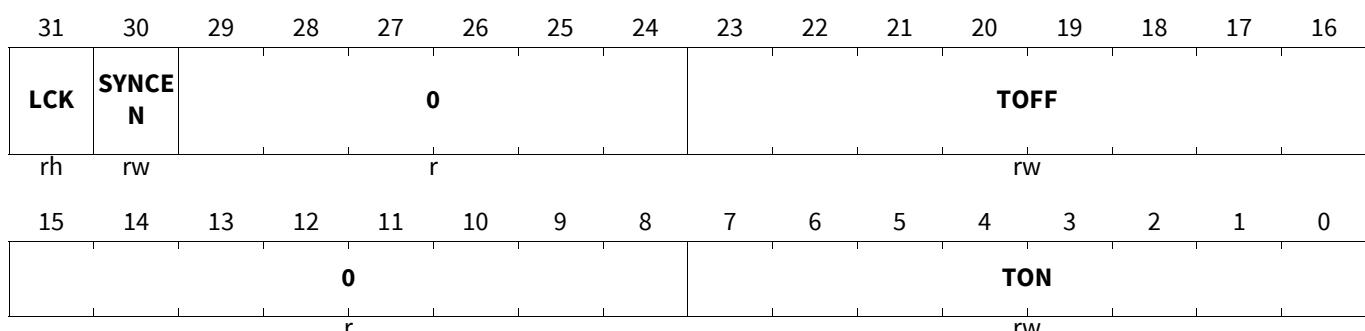
Field	Bits	Type	Description
LCK	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
0	29:4	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 433 Access Mode Restrictions of EVRSDCTRL0 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	SDFREQSPRD	
	rwh	UP	
(default)	r	SDFREQSPRD	
	rh	UP	

**Table 434 Reset Values of EVRSDCTRL0**

Reset Type	Reset Value	Note
LVD Reset	C000 0003 <sub>H</sub>	
Cold PORST	C000 0003 <sub>H</sub>	
After SSW execution	C000 0003 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

**EVRC SD Control Register 1**
**EVRSDCTRL1**
**EVRC SD Control Register 1**
**(010C<sub>H</sub>)**
**Reset Value: Table 436**


## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
TON	7:0	rw	<p><b>Charge Phase length(ton_pulse_len_i)</b>  The charge phase length is defined in nominal 100 MHz clock cycles.</p> <p>Incase SDFREQSPRD = 0 ;  Nominal Switching period without spreading  (SDSWPRDNOM) = TON + TOFF + 18</p> <p>E.g: 1.85 MHz switching frequency case  TON &amp; TOFF = <math>12_H = 18_D</math>; SDFREQSPRD=<math>1_H</math>  <math>\Rightarrow 18 + 18 + 18 + 0 = 54 &gt;</math> SDSWPR (Switching Period)  <math>&gt; 18 + 18 + 18 + 2 = 56</math> clock cycles.  <math>\Rightarrow</math> Actual Switching period with frequency spreading active  (SDSWPRDSPRD) = 1.852 MHz upto 1.786 MHz</p>
TOFF	23:16	rw	<p><b>Discharge Phase length(toff_pulse_len_i)</b>  The discharge phase length in clock cycles. By default, the charge and discharge phase lengths are equal</p>
SYNCEN	30	rw	<p><b>EVRC Synchronization input enable(synci0_en_i) (t.b.d.)</b>  This bitfield enables the input synchronization logic of EVRC SMPS regulator. When set to 1, the DCDC will start to lock to the external synchronization input signal.  This EVRC Synchronization status is indicated in EVRSTAT.SYNCLK status bits.</p> <p><math>0_B</math> Synchronization of EVRC to external input signal is disabled.  <math>1_B</math> Synchronization of EVRC to external input signal is enabled.</p>
LCK	31	rh	<p><b>Lock Status</b>  This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <p><math>0_B</math> The register is unlocked and can be updated  <math>1_B</math> The register is locked and cannot be updated</p>
0	15:8, 29:24	r	<p><b>Reserved</b>  Read as 0; should be written with 0.</p>

Table 435 Access Mode Restrictions of **EVRSCTRL1** sorted by descending priority

Mode Name	Access Mode		Description
LCK = 0	rw	SYNCEN, TOFF, TON	
(default)	r	SYNCEN, TOFF, TON	

Table 436 Reset Values of **EVRSCTRL1**

Reset Type	Reset Value	Note
LVD Reset	8012 0012 <sub>H</sub>	
Cold PORST	8012 0012 <sub>H</sub>	
After SSW execution	8012 0012 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## Power Management System for Low-End (PMSLE)

### EVRC SD Control Register 2

#### EVRSCTRL2

#### EVRC SD Control Register 2

(0110<sub>H</sub>)

Reset Value: [Table 438](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	0	SYNCMUXSEL	0	SYNCHYST			0				SYNCMAXDEV				
rh	r	rw	r	rw			r				rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		DROOPVL					0				DROOPVH				
r		rw					r				rw				

Field	Bits	Type	Description
DROOPVH	4:0	rw	<b>High VDD Limit for Droop request(droopvh_thres_i)</b> This bitfield defines the VDD high voltage limit above which a positive droop request on VDD voltage shall be ignored. VDD Droop High Limit = 712.5 mV + LSB * (SDVOUTSEL+ SDVOUTTRIM+ DROOPVH); LSB = 5 mV
DROOPVL	12:8	rw	<b>Low VDD Limit for Droop request(droopvl_thres_i)</b> This bitfield defines the VDD low voltage limit below which a negative droop request on VDD voltage shall be ignored. VDD Droop Low Limit = 712.5 mV + LSB * (SDVOUTSEL+ SDVOUTTRIM- DROOPVL); LSB = 5 mV
SYNCMAXDEV	20:16	rw	<b>Maximum Deviation of the Synchronization Input Frequency(synci1_maxdev_i)</b> This bitfield defines the maximum allowed frequency deviation of the synchronization input signal frequency from the programmed nominal DCDC switching frequency (EVRSCTRL0.SDFREQ). Violation of limit leads to loss of synchronization. The frequency window is defined as follows $d f_{MAXDEV} = 100 \text{ MHz} * (2 * SYNCMAXDEV) / (SDFREQ^2 + SYNCMAXDEV^2)$ $SYNCMAXDEV = \text{round} [ (100 \text{ MHz} / d f_{MAXDEV}) - \sqrt{(100 \text{ MHz} / d f_{MAXDEV})^2 - SDFREQ^2} ]$
SYNCHYST	26:24	rw	<b>Lock Unlock Hysteresis Window(synci0_hyst_i)</b> This bitfield defines the hysteresis window for synchronization locking and unlocking. The limit is applied to the period counter running at 100 MHz. Upper unlock condition= SDFREQ + SYNCMAXDEV Upper lock condition= SDFREQ + SYNCMAXDEV - SYNCHYST Lower unlock condition = SDFREQ - SYNCMAXDEV Lower lock condition = SDFREQ - SYNCMAXDEV + SYNCHYST $SYNCHYST = \text{round} [ d f_{HYST} * (SDFREQ \pm SYNCMAXDEV)^2 ] / [ d f_{HYST} * (SDFREQ \pm SYNCMAXDEV) + 100 \text{ MHz} ]$

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>SYNCMUXSEL</b>	29:28	rw	<b>Synchronisation Input Multiplexer</b> This bitfield selects synchronisation input either from CCU6 or GTM inputs to be forwarded to EVRC SMPS regulator. 00 <sub>B</sub> Synchronization input open or unconnected. 01 <sub>B</sub> CCU60 COUT63 10 <sub>B</sub> GTM 11 <sub>B</sub> Reserved
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
<b>0</b>	7:5, 15:13, 23:21, 27, 30	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 437 Access Mode Restrictions of [EVRSCTRL2](#) sorted by descending priority**

Mode Name	Access Mode		Description
<b>LCK = 0</b>	rw	DROOPVH, DROOPVL, SYNCHYST, SYNCMAXDEV, SYNCMUXSEL	
(default)	r	DROOPVH, DROOPVL, SYNCHYST, SYNCMAXDEV, SYNCMUXSEL	

**Table 438 Reset Values of [EVRSCTRL2](#)**

Reset Type	Reset Value	Note
LVD Reset	940A 0909 <sub>H</sub>	
Cold PORST	940A 0909 <sub>H</sub>	
After SSW execution	940A 0909 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## Power Management System for Low-End (PMSLE)

### EVRC SD Control Register 3

#### EVRSDCTRL3

#### EVRC SD Control Register 3

(0114<sub>H</sub>)

Reset Value: [Table 440](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>								<b>0</b>							
rh								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							<b>0</b>								
							r								

Field	Bits	Type	Description
<b>MODSEL</b>	0	rw	<b>Manual or Automatic Mode Selection(bypass_mode_i)</b> This bitfield configures manual or automatic selection between SC and BYPass modes.  0 <sub>B</sub> Manual switching between modes enabled - intended for debug use only. BYPSEL is used to disable bypass switches completely (MODSEL=0 and BYPSEL=0) or manually select the number of bypass switches for debugging purposes.  1 <sub>B</sub> Automatic switching between modes enabled. The number of switches is selected based on the present conductance or current output voltage, in combination with the thresholds BYPTHRVLO / BYPTHRVHI for output voltage.
<b>BYPSEL</b>	5:1	rw	<b>Bypass Mode Selection(bypass_sel_i)</b> This bitfield selects the number of switches to be activated during manual bypass mode (MODSEL=0).
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
<b>0</b>	30:6	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 439 Access Mode Restrictions of EVRSDCTRL3 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	BYPSEL, MODSEL	
(default)	r	BYPSEL, MODSEL	

## Power Management System for Low-End (PMSLE)

**Table 440 Reset Values of EVRSDCTRL3**

Reset Type	Reset Value	Note
LVD Reset	8000 0001 <sub>H</sub>	
Cold PORST	8000 0001 <sub>H</sub>	
After SSW execution	8000 0001 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVRC SD Control Register 4

**EVRSDCTRL4**

**EVRC SD Control Register 4 (0118<sub>H</sub>)**

**Reset Value: Table 442**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	<b>0</b>	<b>ZEROBIN</b>	<b>0</b>												
rh	r	rw	r												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>OLEN</b>	<b>0</b>									<b>0</b>					
rw	r			rw						r			rw		

Field	Bits	Type	Description
<b>SDVOKLVL</b>	5:0	rw	<b>Voltage OK Circuit Configuration(config1_voltok_i)</b> This bit field configures the output voltage level for voltage OK configuration threshold to switch from start-up to closed loop operation phase. TC [0:1] : Time constant Filter implementation VTHR [5:2] : Voltage threshold level
<b>SDLUT</b>	13:8	rw	<b>Non linearity slope and threshold(config0_lut_i)</b> This bit field configures the non linearity slope and threshold after ADC. [5:4] - Slope ; [3:0] - Threshold E.g. X <sub>V</sub> - X <sub>H</sub> - X <sub>D</sub>
<b>OLEN</b>	15	rw	<b>Open Loop Operation Enable(open_loop_op_i)</b> This bit field activates open or closed loop operation. 0 <sub>B</sub> Closed Loop operation. 1 <sub>B</sub> Open Loop operation.
<b>SDOLCON</b>	26:16	rw	<b>Initial Conductance during Start-up(open_loop_init_i)</b> This bit field configures the value of Conductance for Start-up and open loop operation. Activates parallel switches to increase conductance.
<b>ZEROBIN</b>	29:28	rw	<b>Stabilization strength Zero Error Bin(zero_bin_i)</b> This bit field configures the stabilization strength during zero crossing to avoid limit cycles. 00 <sub>B</sub> zero bin inactive. 01 <sub>B</sub> zero bin active. strength factor 1/2 10 <sub>B</sub> zero bin active. strength factor 1/4 11 <sub>B</sub> zero bin active. strength factor 1/8

## **Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
LCK	31	rh	<p><b>Lock Status</b></p> <p>This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <p><math>0_B</math> The register is unlocked and can be updated  <math>1_B</math> The register is locked and cannot be updated</p>
0	7:6, 14, 27, 30	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 441 Access Mode Restrictions of EVRSDCTRL4 sorted by descending priority**

<b>Mode Name</b>	<b>Access Mode</b>		<b>Description</b>
<b>LCK = 0</b>	rw	OLEN, SDLUT, SDOLCON, SDVOKLVL, ZEROBIN	
(default)	r	OLEN, SDLUT, SDOLCON, SDVOKLVL, ZEROBIN	

**Table 442 Reset Values of EVRSDCTRL4**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
LVD Reset	A000 2209 <sub>H</sub>	
Cold PORST	A000 2209 <sub>H</sub>	
After SSW execution	A000 2209 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## **EVRC SD Control Register 5**

EVRSDCTRL5

## **EVRC SD Control Register 5**

(011C<sub>H</sub>)

### **Reset Value: Table 444**

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK						0						STCONDEC	STCONLIMIN C	STCO NLIME N		
rh					r							rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
STTH32ROFF		STEN1 6R32R	STSPEED5V		STDT		STDTE N	STSPEED3V3								
rw		rw	rw		rw		rw	rw				rw	rw		rw	rw

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>STSPEED3V3</b>	1:0	rw	<b>3V3 Integration Start-up coefficient(startup0_speed3v3_i)</b> This bitfield selects the integration coefficient during start-up for VEXT = 3.3V. 00 <sub>B</sub> SD33I = 2 01 <sub>B</sub> SD33I = 3 10 <sub>B</sub> SD33I = 4 11 <sub>B</sub> SD33I = 5
<b>STDTEEN</b>	2	rw	<b>Dead time Enable during start-up(startup0_dten_i)</b> This bitfield activates dead time during start-up phase. Dead time is fixed to the value selected until start-up is finished and then reduced by 1 every switching cycle 0 <sub>B</sub> Dead time at start-up inactive 1 <sub>B</sub> Dead time at start-up = STDT
<b>STDT</b>	7:3	rw	<b>Dead time value during start-up(startup0_dt_i)</b> This bitfield configures the dead time value during start-up. It is multiplied by 2 inside the controller and half is subtracted from charge time and half from discharge time.
<b>STSPEED5V</b>	9:8	rw	<b>5VIntegration Start-up coefficient(startup1_speed5v_i)</b> This bitfield selects the integration coefficient during start-up for VEXT = 5V. 00 <sub>B</sub> SD5I = 1 01 <sub>B</sub> SD5I = 2 10 <sub>B</sub> SD5I = 3 11 <sub>B</sub> SD5I = 4
<b>STEN16R32R</b>	10	rw	<b>Subswitch selection during start-up(startup1_en16r32r_i)</b> This bitfield selects the usage 16R and 32R sub-switches in SW8 and SW9 during start-up. 0 <sub>B</sub> 16R and 32R sub-switches not used during start-up. 1 <sub>B</sub> 16R and 32R sub-switches are used instead of two R sub-switches.
<b>STTH32ROFF</b>	15:11	rw	<b>Threshold to turn off 32R sub-switch during start-up(startup1_th32roff_i)</b> This bitfield specifies the threshold to turn off 32R sub-switch. Dead time is used as a counter. Once dead time after start-up reaches this threshold, the 32R switch is removed from operation.
<b>STCONLIMEN</b>	16	rw	<b>Conductance Limitation Enable(condlim2_en_i)</b> This bitfield enables conductance limitation. 0 <sub>B</sub> Conductance limitation disabled 1 <sub>B</sub> Conductance limitation enabled
<b>STCONLIMINC</b>	18:17	rw	<b>Start-up Conductance Limit Increment Factor(condlim2_startinc_i)</b> This bitfield selects the incrementing factor for conductance limit during start-up 00 <sub>B</sub> Conductance Limit = Nominal Conductance Limit 01 <sub>B</sub> Conductance Limit = Nominal Conductance Limit * 1.5 10 <sub>B</sub> Conductance Limit = Nominal Conductance Limit * 1.25 11 <sub>B</sub> Conductance Limit = Nominal Conductance Limit * 1.125

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>STCONDEC</b>	20:19	rw	<b>Conductance decrementing factor during start-up(startup3_conddec_i)</b> This bitfield selects the decrementing factor for Conductance during transition from start-up to normal operation. 00 <sub>B</sub> Conductance = Nominal Conductance * 0.5 01 <sub>B</sub> Conductance = Nominal Conductance * 0.75 10 <sub>B</sub> Conductance = Nominal Conductance * 0.875 11 <sub>B</sub> Conductance = Nominal Conductance * 0.9375
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
<b>0</b>	30:21	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 443 Access Mode Restrictions of EVRSDCTRL5 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	STCONDEC, STCONLIMEN, STCONLIMINC, STDT, STDTEN, STEN16R32R, STSPEED3V3, STSPEED5V, STTH32ROFF	
(default)	r	STCONDEC, STCONLIMEN, STCONLIMINC, STDT, STDTEN, STEN16R32R, STSPEED3V3, STSPEED5V, STTH32ROFF	

**Table 444 Reset Values of EVRSDCTRL5**

Reset Type	Reset Value	Note
LVD Reset	801B 7566 <sub>H</sub>	
Cold PORST	801B 7566 <sub>H</sub>	
After SSW execution	801B 7566 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## Power Management System for Low-End (PMSLE)

### EVRC SD Control Register 6

#### EVRSDCTRL6

#### EVRC SD Control Register 6

(0120<sub>H</sub>)

Reset Value: [Table 446](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK			0			SYNCDIVFAC			SYNCSPONT			0		SYNCLKOPT	
rh			r			rw			rw			r		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
															r

Field	Bits	Type	Description
SYNCLKOPT	16	rw	<b>First synchronization option selection(synci0_lockopt_i)</b> $0_B$ Fast synchronization (typically 1-2 sw cycles) $1_B$ Slow synchronization (typically 4-8 sw cycles) (default)
SYNCSPONT	23:20	rw	<b>Synchronization point(synci2_spoint_i)</b> T(dcdc period) - SPOINT will be synchronized to the rising edge of sd_sync_in.
SYNCDIVFAC	26:24	rw	<b>Switching frequency division factor for external synchronization(sd_syncdiv_i)</b> This bit field defines the divider factor for the SMPS switching output to generate DCDCSYNCO output to synchronize external EVRC regulator to the internal EVRC regulator. The signal is routed to pin if enabled via PMSWCR5.DCDCSYNCO bit. All other combinations are reserved. $000_B$ $f_{DCDCSYNCO} = f_{DCDC}$ . The actual duty cycle is routed. $001_B$ $f_{DCDCSYNCO} = f_{DCDC}/2$ . Duty cycle is constant at 50%. $010_B$ $f_{DCDCSYNCO} = f_{DCDC}/4$ . Duty cycle is constant at 50%. $011_B$ $f_{DCDCSYNCO} = f_{DCDC}/8$ . Duty cycle is constant at 50%. $100_B$ $f_{DCDCSYNCO} = f_{DCDC}/16$ . Duty cycle is constant at 50%. $101_B$ $f_{DCDCSYNCO} = f_{DCDC}/32$ . Duty cycle is constant at 50%. $110_B$ $f_{DCDCSYNCO} = f_{DCDC}/64$ . Duty cycle is constant at 50%. $111_B$ $f_{DCDCSYNCO} = f_{DCDC}/128$ . Duty cycle is constant at 50%.
LCK	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. $0_B$ The register is unlocked and can be updated $1_B$ The register is locked and cannot be updated
0	15:0, 19:17, 30:27	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System for Low-End (PMSLE)

**Table 445 Access Mode Restrictions of EVRSDCTRL6 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	SYNCDIVFAC, SYNCLOCKPT, SYNCSPPOINT	
(default)	r	SYNCDIVFAC, SYNCLOCKPT, SYNCSPPOINT	

**Table 446 Reset Values of EVRSDCTRL6**

Reset Type	Reset Value	Note
LVD Reset	8081 0000 <sub>H</sub>	
Cold PORST	8081 0000 <sub>H</sub>	
After SSW execution	8081 0000 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## EVRC SD Control Register 7

**EVRSDCTRL7**

**EVRC SD Control Register 7** (0124<sub>H</sub>) Reset Value: [Table 448](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	0	<b>FBADC LSB</b>		0			<b>FBADCNS</b>	<b>FBADCLPF</b>		0		<b>FBADCBLNK</b>			
rh	r	rw		r			rw		rw	rw		r		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			<b>FBADCSMP</b>									<b>FBADCOFFS</b>			
r			rw									rw			

Field	Bits	Type	Description
<b>FBADCOFFS</b>	7:0	rw	<b>Feedback Converted Counter Value Offset(adctr2_offset_i)</b> This bitfield configures the offset of the converted counter value of the feedback ADC measuring the core voltage. This configures the offset to the default offset 0.7125V. FBADCOFFS RANGE = -128 to 127 LSB ; LSB = 5 mV
<b>FBADCSMP</b>	13:8	rw	<b>FB ADC Sampling period(adctr_smpthr_i)</b> This bitfield configures the sampling period in 100 MHz clock cycles for the feedback ADC measuring the core voltage. FBADC clock = f BACK / FBADCSMP (unsigned).
<b>FBADCBLNK</b>	17:16	rw	<b>FB ADC Blanked Samples Number(adctr_blank_i)</b> This bitfield configures the number of feedback ADC samples that are blanked during charge / discharge phase transitions to minimise switching noise influence.This is implemented in tracking ADC counter.

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>FBADCLPF</b>	21:20	rw	<b>FB ADC Counter LPF Coefficient(adctr_lpf_i)</b> This bit field configures the coefficient of the Low Pass Filter of the feedback ADC counter value measuring the core voltage. $y[k] = \{y[k-1] * (1-a\} + \{x[k] * a\}$ ; $y[k]$ is filter output; $x[k]$ is ADC output. $a = \{1 / (2^LPF)\}$ . If LPF = 0, the filter output is the same as ADC output. 00 <sub>B</sub> no filtering 01 <sub>B</sub> 1/2 10 <sub>B</sub> 1/4 11 <sub>B</sub> 1/8
<b>FBADCNS</b>	23:22	rw	<b>FB ADC Noise shaper setting(noise_shape_i)</b> This bit field configures the noise shaper of feedback ADC and whether the quantization error is considered. 00 <sub>B</sub> No noise shaping 01 <sub>B</sub> First order noise shaping 10 <sub>B</sub> Second order noise shaping 11 <sub>B</sub> Reserved
<b>FBADCLSB</b>	29	rw	<b>FB ADC LSB for Error Computation(adctr_lsb_i)</b> This bitfield configures the LSB of the feedback ADC counter value used for the error computation. 0 <sub>B</sub> 5 mV 1 <sub>B</sub> 10 mV (default)
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
<b>0</b>	15:14, 19:18, 28:24, 30	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 447 Access Mode Restrictions of EVRSDCTRL7 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	FBADCBLNK, FBADCLPF, FBADCLSB, FBADCNS, FBADCOFFS, FBADCSMP	
(default)	r	FBADCBLNK, FBADCLPF, FBADCLSB, FBADCNS, FBADCOFFS, FBADCSMP	

**Table 448 Reset Values of EVRSDCTRL7**

Reset Type	Reset Value	Note
LVD Reset	A061 0400 <sub>H</sub>	

## Power Management System for Low-End (PMSLE)

**Table 448 Reset Values of EVRSDCTRL7 (cont'd)**

Reset Type	Reset Value	Note
Cold PORST	A061 0400 <sub>H</sub>	
After SSW execution	A061 0400 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

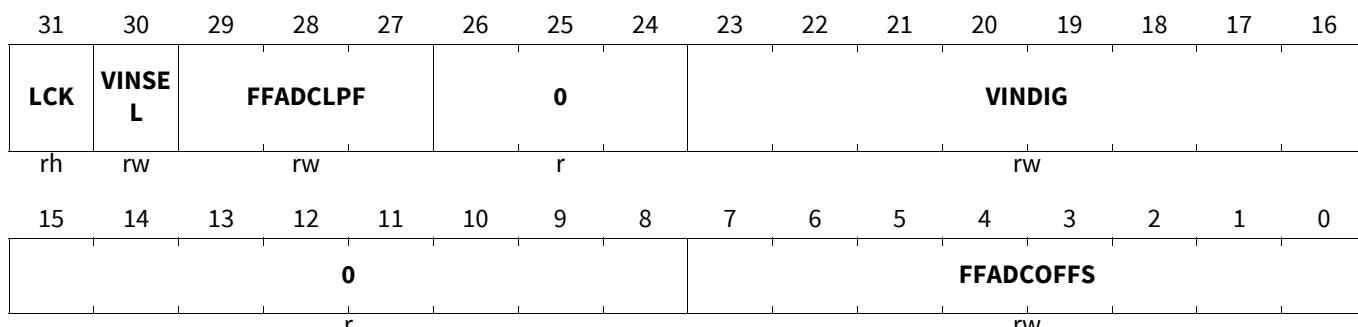
### EVRC SD Control Register 8

**EVRSDCTRL8**

**EVRC SD Control Register 8**

(0128<sub>H</sub>)

**Reset Value: Table 450**



Field	Bits	Type	Description
<b>FFADCOFFS</b>	7:0	rw	<b>Feed Forward Converted Counter Value Offset(ffadc_offset_i)</b> This bit field configures the offset of the converted counter value of the feed forward ADC measuring the input VEXT voltage. Signed value [-128...127], LSB = 20 mV
<b>VINDIG</b>	23:16	rw	<b>VEXT input voltage digital value(ffadc3_vindig_i)</b> This bit field specifies a fixed input voltage value which can be used by PI controller instead of FF ADC result. VIN = [LSB * VINDIG + 1.050] V ; LSB = 20 mV
<b>FFADCLPF</b>	29:27	rw	<b>FF ADC Counter LPF Coefficient(ffadc_lpf_i)</b> This bit field configures the coefficient of the Low Pass Filter of the feed-forward ADC counter value measuring the input VEXT voltage. $y[k] = \{y[k-1] * (1-a)\} + \{x[k] * a\}$ ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}. If LPF = 0, the filter output is the same as ADC output. 000 <sub>B</sub> no filtering 001 <sub>B</sub> 1/2 010 <sub>B</sub> 1/4 011 <sub>B</sub> 1/8
<b>VINSEL</b>	30	rw	<b>VEXT Input Voltage Source Selection(ffadc0_inputsel_i)</b> This bit field selects the source of input voltage information for PI controller 0 <sub>B</sub> FF ADC not used. 1 <sub>B</sub> VINDIG value taken.

**Power Management System for Low-End (PMSLE)**

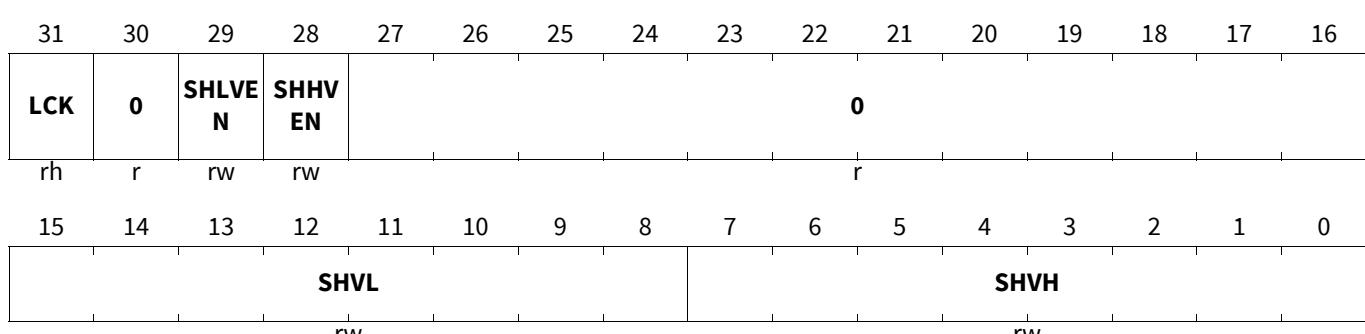
Field	Bits	Type	Description
LCK	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
0	15:8, 26:24	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 449 Access Mode Restrictions of EVRSDCTRL8 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	FFADCLPF, FFADCOFFS, VINDIG, VINSEL	
(default)	r	FFADCLPF, FFADCOFFS, VINDIG, VINSEL	

**Table 450 Reset Values of EVRSDCTRL8**

Reset Type	Reset Value	Note
LVD Reset	9070 0000 <sub>H</sub>	
Cold PORST	9070 0000 <sub>H</sub>	
After SSW execution	9070 0000 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

**EVRC SD Control Register 9**
**EVRSDCTRL9**
**EVRC SD Control Register 9**
(012C<sub>H</sub>)Reset Value: [Table 452](#)

Field	Bits	Type	Description
SHVH	7:0	rw	<b>Short to High Voltage Threshold(short1_thr_i)</b> The counter value of the tracking ADC is compared against SHVH. VOUT + SHVH x 5 mV
SHVL	15:8	rw	<b>Short to Low Voltage Threshold(short2_thr_i)</b> The counter value of the tracking ADC is compared against SHVL. VOUT + SHVL x 5 mV

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>SHHVEN</b>	28	rw	<b>Short to High Detection Enable(short3_shhven_i)</b> 0 <sub>B</sub> Short to High Detection is disabled 1 <sub>B</sub> Short to High Detection is enabled
<b>SHLVEN</b>	29	rw	<b>Short to Low Detection Enable(short3_shlven_i)</b> 0 <sub>B</sub> Short to Low Detection is disabled 1 <sub>B</sub> Short to Low Detection is enabled
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
<b>0</b>	27:16, 30	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 451 Access Mode Restrictions of EVRSDCTRL9 sorted by descending priority**

Mode Name	Access Mode		Description
<b>LCK = 0</b>	rw	SHHVEN, SHLVEN, SHVH, SHVL	
(default)	r	SHHVEN, SHLVEN, SHVH, SHVL	

**Table 452 Reset Values of EVRSDCTRL9**

Reset Type	Reset Value	Note
LVD Reset	8000 4040 <sub>H</sub>	
Cold PORST	8000 4040 <sub>H</sub>	
After SSW execution	8000 4040 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

**EVRC SD Control Register 10**
**EVRSDCTRL10**
**EVRC SD Control Register 10** **Reset Value: Table 454**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	<b>0</b>	<b>DTOV5VBASE</b>					<b>0</b>	<b>DTOV3V3BASE</b>							
rh	r							r					rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							<b>DTOVERRMIN</b>	<b>DTOVERRREN</b>	<b>0</b>	<b>DTLOCNCONMIN</b>					<b>DTLOCONEN</b>
							rw	r					rw		rw

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>DTLOCONEN</b>	0	rw	<b>Enable Dead time at low conductance(dtlocon_en_i)</b> This bitfield activates the incorporation of dead time between charge and discharge phase when conductance is below 4. The charge and discharge phase are reduced by half of dead time value each.
<b>DTLOCONMIN</b>	6:1	rw	<b>Minimum Charge + Discharge time after Dead time(dtlocon_mintime_i)</b> This bitfield defines the minimum number of SDSWPRDNOM clock cycles for charge + discharge phase after dead time introduction. The minimum setting (duration of charge+discharge phase) is 4 clock cycles.
<b>DTOVERREN</b>	8	rw	<b>Enable Dead time on VDD output voltage error(dtov_en_i)</b> This bitfield activates the incorporation of dead time when VDD output voltage error is above / below threshold DTOVERRMIN. DTOV = (DTOV3V3 / DTOV5V + increment) An increment of 2 clock cycles of dead time is made with respective increase of output voltage error of greater than 3 LSB.
<b>DTOVERRMIN</b>	12:9	rw	<b>Dead time VDD Output Voltage Error Threshold(dtov_thr_i)</b> This bitfield defines the VDD output voltage error threshold beyond which dead time is introduced. The threshold should not be below 3 because error upto 2 can happen during normal operation.
<b>DTOV3V3BASE</b> <b>E</b>	20:16	rw	<b>Dead time 3V3 base value for Output Voltage Error(dtov3v3_base_i)</b> This bitfield defines the base dead time value of SDSWPRDNOM clock cycles introduced when VDD output voltage error is above / below threshold DTOVERRMIN during over / undershoots. DTOV3V3 is selected for nominal VEXT = 3.3V case.
<b>DTOV5VBASE</b>	28:24	rw	<b>Dead time 5V base value for Output Voltage Error(dtov5v_base_i)</b> This bitfield defines the base dead time value of SDSWPRDNOM clock cycles introduced when VDD output voltage error is above / below threshold DTOVERRMIN during over / undershoots. DTOV5V is selected for nominal VEXT = 5V case.
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. $0_B$ The register is unlocked and can be updated $1_B$ The register is locked and cannot be updated
<b>0</b>	7, 15:13, 23:21, 30:29	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System for Low-End (PMSLE)

**Table 453 Access Mode Restrictions of EVRSDCTRL10 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	DTLOCONEN, DTLOCONMIN, DTOV3V3BASE, DTOV5VBASE, DTOVERREN, DTOVERRMIN	
(default)	r	DTLOCONEN, DTLOCONMIN, DTOV3V3BASE, DTOV5VBASE, DTOVERREN, DTOVERRMIN	

**Table 454 Reset Values of EVRSDCTRL10**

Reset Type	Reset Value	Note
LVD Reset	930C 0719 <sub>H</sub>	
Cold PORST	930C 0719 <sub>H</sub>	
After SSW execution	930C 0719 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### EVRC SD Coefficient Register 0

#### EVRSDCOEFF0

#### EVRC SD Coefficient Register 0

(0148<sub>H</sub>)

Reset Value: [Table 456](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>					<b>0</b>				<b>SD5P</b>			<b>SD5I</b>			
rh				r					rw			rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					<b>0</b>				<b>SD33P</b>			<b>SD33I</b>			
				r					rw			rw			

Field	Bits	Type	Description
<b>SD33I</b>	3:0	rw	<b>I Coefficient(coeff_i_3v3_i)</b> I control parameter for the PI regulator (VEXT = 3.3V).
<b>SD33P</b>	7:4	rw	<b>P Coefficient(coeff_p_3v3_i)</b> P control parameter for the PI regulator (VEXT = 3.3V).
<b>SD5I</b>	19:16	rw	<b>I Coefficient(coeff_i_5v0_i)</b> I control parameter for the PI regulator (VEXT = 5V).
<b>SD5P</b>	23:20	rw	<b>P Coefficient(coeff_p_5v0_i)</b> P control parameter for the PI regulator (VEXT = 5V).
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
0	15:8, 30:24	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 455 Access Mode Restrictions of EVRSDCOEFF0 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	SD33I, SD33P, SD5I, SD5P	
(default)	r	SD33I, SD33P, SD5I, SD5P	

**Table 456 Reset Values of EVRSDCOEFF0**

Reset Type	Reset Value	Note
LVD Reset	8052 0083 <sub>H</sub>	
Cold PORST	8052 0083 <sub>H</sub>	
After SSW execution	8052 0083 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

**EVRC SD Coefficient Register 1**
**EVRSDCOEFF1**
**EVRC SD Coefficient Register 1**
**(014C<sub>H</sub>)**
**Reset Value: Table 458**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	0	<b>CHNGHYST</b>				<b>PILOV ADPT</b>	<b>PILOV ADEN</b>								<b>PILOV5V</b>
rh	r		rw			rw	rw								rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			<b>PILOV3V3</b>						<b>PICHNGTHR</b>						
			rw						rw						

Field	Bits	Type	Description
<b>PICHNGTHR</b>	7:0	rw	<b>PI Coefficients Change VEXT Threshold(pichng_thr_i)</b> This bitfield reflects VEXT input voltage threshold for switching of PI coefficients from 5V to 3.3V based on the FF-ADC VEXT measurement value. The SD5x values are used incase of VEXT = 5V supply topology and The SD33x values are used incase of VEXT = 3.3V supply topology respectively. VTHR = [LSB * PICHNGTHR+ 1.050] V ; LSB = 20 mV E.g. 4.050 V - 96 <sub>HH</sub> - 150 <sub>DD</sub>
<b>PILOV3V3</b>	15:8	rw	<b>3V3 PI Coefficients Adaptation Threshold at Low VEXT(pilov3v3_thr_i)</b> This bitfield reflects the PI coefficient adaption VEXT input voltage threshold at 3.3V - 10% based on the FF-ADC VEXT measurement value. VTHR = [LSB * PILOV3V3+ 1.050] V ; LSB = 20 mV E.g. 3.150 V - 69 <sub>H</sub> - 105 <sub>D</sub>

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>PILOV5V</b>	23:16	rw	<b>5V PI Coefficients Adaptation Threshold at Low VEXT(pilov5v0_thr_i)</b> This bitfield reflects the PI coefficient adaption VEXT input voltage threshold at 5V - 10% based on the FF-ADC VEXT measurement value. VTHR = [LSB * PILOV5V+ 1.050] V ; LSB = 20 mV E.g. 4.610 V - B2 <sub>H</sub> - 178 <sub>D</sub>
<b>PILOVADEN</b>	24	rw	<b>PI self adaptation enable(pi_self_adapt_i)</b> This bitfield activates self adaptation of PI coefficients for different load currents based on conductance value. 0 <sub>B</sub> no self adaptation active 1 <sub>B</sub> self adaptation active
<b>PILOVADPT</b>	25	rw	<b>PI adaptation coefficient at Low VEXT(piself_lovadapt_i)</b> This bitfield configures self adaptation coefficient for PI parameters for -10% input VEXT voltage. 0 <sub>B</sub> no adaptation 1 <sub>B</sub> I coefficient is increased by 1 if VEXT is lower than the selected threshold
<b>CHNGHYST</b>	29:26	rw	<b>Hysteresis for VEXT PI parameter change(piself_chnghyst_i)</b> This bitfield configures the hysteresis for PI change between 3.3V / 5V sets based on the FF-ADC VEXT measurement value. Change to 5V coefficient set if VIN > [LSB * (PICHNGTHR+CHNGHYST)+1.050] V ; LSB = 20 mV Change to 3.3V coefficient set if VIN < [LSB * (PICHNGTHR-CHNGHYST)+1.050] V ; LSB = 20 mV
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
<b>0</b>	30	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 457 Access Mode Restrictions of EVRSDCOEFF1 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	CHNGHYST, PICHNGTHR, PILOV3V3, PILOV5V, PILOVADEN, PILOVADPT	
(default)	r	CHNGHYST, PICHNGTHR, PILOV3V3, PILOV5V, PILOVADEN, PILOVADPT	

**Table 458 Reset Values of EVRSDCOEFF1**

Reset Type	Reset Value	Note
LVD Reset	97B2 6996 <sub>H</sub>	

## Power Management System for Low-End (PMSLE)

**Table 458 Reset Values of EVRSDCOEFF1 (cont'd)**

Reset Type	Reset Value	Note
Cold PORST	97B2 6996 <sub>H</sub>	
After SSW execution	97B2 6996 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

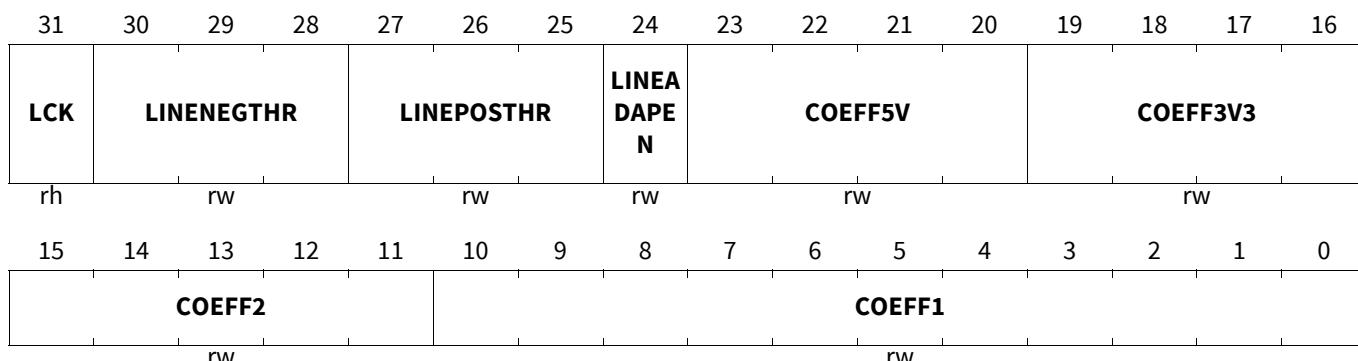
### EVRC SD Coefficient Register 2

#### EVRSDCOEFF2

#### EVRC SD Coefficient Register 2

(0150<sub>H</sub>)

Reset Value: [Table 460](#)



Field	Bits	Type	Description
COEFF1	10:0	rw	<b>Coefficient 1(condlimx_coeff1x_i)</b> This bitfield specifies the Coefficient 1. $COEFF1 = 50 * (Load * Rsw) / SWp$ , where: Load = load current limit in [A] Rsw = resistance of one 8R sub-switch in [Ohm] SWp = switching frequency penalty: $SWp = (Ton + Toff + 6) / (Ton + Toff + 18)$
COEFF2	15:11	rw	<b>Coefficient 2(condlim1_coeff2_i)</b> This bitfield specifies the Coefficient 2. $COEFF2 = 50 * (Load * Rbond) / SWp$ , where: Load = load current limit in [A] Rsw = bonding wire resistance + parasitics in [Ohm] SWp = switching frequency penalty: $SWp = (Ton + Toff + 6) / (Ton + Toff + 18)$
COEFF3V3	19:16	rw	<b>3V3 Line Adaptation Coefficient(lcoeff_coeff3v3_i)</b> This bitfield specifies the line adaptation coefficient for input voltage in 3.3V range. Integrator = Integrator + $2^{(7+COEFF3V3)} * FF\_ADC\_error$ FF_ADC_error = FF ADC result difference to the one read out at previous switching cycle.

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>COEFF5V</b>	23:20	rw	<b>5V Line Adaptation Coefficient(lcoeff_coeff5v_i)</b> This bitfield specifies the line adaptation coefficient for input voltage in 5V range. $\text{Integrator} = \text{Integrator} + 2^{(7+\text{COEFF5V})} * \text{FF\_ADC\_error}$ $\text{FF\_ADC\_error} = \text{FF ADC result difference to the one read out at previous switching cycle.}$
<b>LINEADAPEN</b>	24	rw	<b>Integrator Adaptation Enable on Line Jump(lineadap_en_i)</b> This bitfield configures the Enable/Disable adaptation of Integrator during input voltage transition. 0 <sub>B</sub> Adaptation disabled 1 <sub>B</sub> Adaptation enabled
<b>LINEPOSTHR</b>	27:25	rw	<b>Positive Threshold for VEXT line change(lineadap_posthr_i)</b> This bitfield specifies the positive threshold for input voltage cycle to cycle difference (i.e. for the FF ADC result difference to the one read out at previous switching cycle). LSB = 20 mV
<b>LINENEGTHR</b>	30:28	rw	<b>Negative Threshold for VEXT line change(lineadap_negthr_i)</b> This bitfield specifies the negative threshold for input voltage cycle to cycle difference (i.e. for the FF ADC result difference to the one read out at previous switching cycle). LSB = 20 mV
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated

**Table 459 Access Mode Restrictions of EVRSDCOEFF2 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	COEFF1, COEFF2, COEFF3V3, COEFF5V, LINEADAPEN, LINENEGTHR, LINEPOSTHR	
(default)	r	COEFF1, COEFF2, COEFF3V3, COEFF5V, LINEADAPEN, LINENEGTHR, LINEPOSTHR	

**Table 460 Reset Values of EVRSDCOEFF2**

Reset Type	Reset Value	Note
LVD Reset	C924 8BD9 <sub>H</sub>	
Cold PORST	C924 8BD9 <sub>H</sub>	
After SSW execution	C924 8BD9 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

## Power Management System for Low-End (PMSLE)

### EVRC SD Coefficient Register 3

#### EVRSDCOEFF3

#### EVRC SD Coefficient Register 3

(0154<sub>H</sub>)

Reset Value: [Table 462](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>	<b>0</b>		<b>SDDIGIN2</b>								<b>SDDIGIN1</b>				
rh	r														rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>SDDIGINO</b>								<b>BYPONTHR</b>							
															rw

Field	Bits	Type	Description
<b>BYPONTHR</b>	7:0	rw	<b>Bypass mode On VEXT threshold(bponthr_thr_i)</b> This bitfield specifies the VEXT threshold for bypass use. If VEXT is above the threshold, bypass switches will not be turned on. The results from feed forward VEXT primary ADC is used. VIN = [LSB * ADCSWDV+ 1.050] V ; LSB = 20 mV E.g. x V - 78 <sub>H</sub> - x <sub>D</sub>
<b>SDDIGINO</b>	15:8	rw	<b>Digital Control(config3_i)</b> This bitfield is reserved.
<b>SDDIGIN1</b>	20:16	rw	<b>Digital Control(config4_i)</b> This bitfield is reserved.
<b>SDDIGIN2</b>	28:21	rw	<b>Digital Control(config2_i)</b> This bitfield is reserved.
<b>LCK</b>	31	rh	<b>Lock Status</b> This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 <sub>B</sub> The register is unlocked and can be updated 1 <sub>B</sub> The register is locked and cannot be updated
<b>0</b>	30:29	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 461 Access Mode Restrictions of EVRSDCOEFF3 sorted by descending priority**

Mode Name	Access Mode		Description
LCK = 0	rw	BYPONTHR, SDDIGINO, SDDIGIN1, SDDIGIN2	
(default)	r	BYPONTHR, SDDIGINO, SDDIGIN1, SDDIGIN2	

## Power Management System for Low-End (PMSLE)

**Table 462 Reset Values of EVRSDCOEFF3**

Reset Type	Reset Value	Note
LVD Reset	8780 00A3 <sub>H</sub>	
Cold PORST	8780 00A3 <sub>H</sub>	
After SSW execution	8780 00A3 <sub>H</sub>	The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.

### 12.3.2.3 Die Temperature Sensor Registers

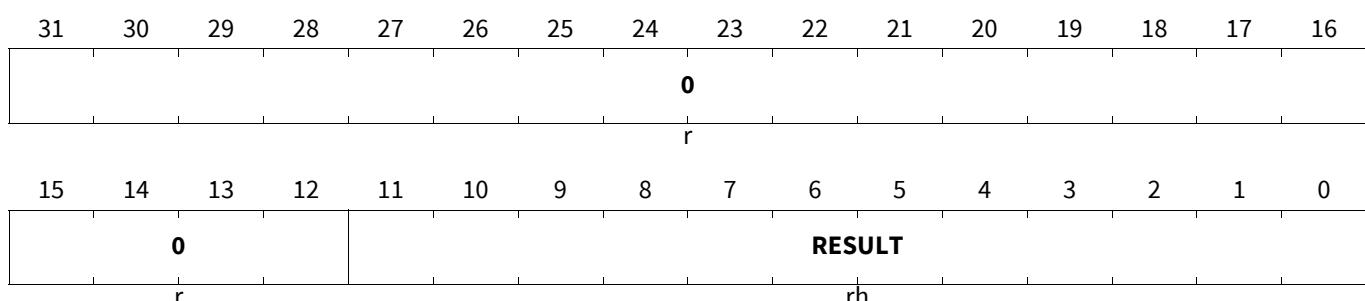
#### Die Temperature Sensor Status Register

DTSSTAT

Die Temperature Sensor Status Register

(01C0<sub>H</sub>)

Reset Value: [Table 463](#)



Field	Bits	Type	Description
<b>RESULT</b>	11:0	rh	<b>Result of the DTS Measurement</b> This bit field shows the result of the DTS measurement. The value given is directly related to the die temperature and can be evaluated using the following formula. $T (\text{°C}) = [\text{RESULT} / \text{G}_{\text{nom}}] - 273.15$ $T (\text{°K}) = [\text{RESULT}] / \text{G}_{\text{nom}}$ $\text{RESULT} = \text{G}_{\text{nom}} * \{T (\text{°C}) + 273.15\} = \text{G}_{\text{nom}} * T (\text{°K})$ $\text{G}_{\text{nom}} = 7.505$
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0.

**Table 463 Reset Values of DTSSTAT**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Cold PORST	0000 0000 <sub>H</sub>	

## Power Management System for Low-End (PMSLE)

### Die Temperature Sensor Limit Register

#### DTSLIM

#### Die Temperature Sensor Limit Register

(01C8<sub>H</sub>)

Reset Value: [Table 465](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>UOF</b>	<b>SLCK</b>	<b>0</b>	<b>UPPER</b>												
rwh	rw	r													rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>LLU</b>	<b>0</b>	<b>LOWER</b>													
rwh	r														rw

Field	Bits	Type	Description
<b>LOWER</b>	11:0	rw	<b>Lower Limit</b> This bit field defines the lower limit of the DTS temperature check. The DTS measurement result is compared against this value and if the measurement result is less than or equal to the configured LOWER bitfield value; flag LLU is set.
<b>LLU</b>	15	rwh	<b>Lower Limit Underflow</b> When this bit is set, a HSM temperature underflow trigger is generated. When this bit is set the related SMU DTS alarm trigger is generated. This bit has to be written with zero in order to clear it. Writing a one has no effect. This bit is set when a DTS measurement is finished and the result is below the lower limit (i.e. DTSLIM.LOWER). 0 <sub>B</sub> No temperature underflow was detected 1 <sub>B</sub> A temperature underflow was detected
<b>UPPER</b>	27:16	rw	<b>Upper Limit</b> This bit field defines the upper limit of the DTS temperature check. The DTS measurement result is compared against this value and if the measurement result is greater than or equal to the configured UPPER bitfield value; flag UOF is set.
<b>SLCK</b>	30	rw	<b>HSM Security Lock</b> If this bit is set, all other bits in this register can no longer be written. Write requests to other bits when SLCK is set will trigger an SLCK access error alarm. This bit can not be cleared by software. SLCK bit can only be set by an access from the HSM master (TAG = 000011 <sub>B</sub> ). A set operation performed by any other master or software is ignored and the bit is kept as cleared. 0 <sub>B</sub> No lock active 1 <sub>B</sub> Lock is active

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>UOF</b>	31	rwh	<p><b>Upper Limit Overflow</b></p> <p>When this bit is set, a HSM temperature overflow trigger is generated.</p> <p>When this bit is set, the related SMU DTS alarm trigger is generated.</p> <p>This bit has to be written with zero in order to clear it. Writing a one has no effect.</p> <p>This bit is set when a DTS measurement is finished and the result is exceeding the upper limit (i.e. DTSLIM.UPPER).</p> <p>0<sub>B</sub> No temperature overflow was detected 1<sub>B</sub> A temperature overflow was detected</p>
<b>0</b>	14:12, 29:28	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 464 Access Mode Restrictions of DTSLIM sorted by descending priority**

Mode Name	Access Mode		Description
<b>SLCK = 0</b>	rw	LOWER, UPPER	
	rwh	LLU, UOF	
(default)	r	LOWER, SLCK, UPPER	
	rh	LLU, UOF	

**Table 465 Reset Values of DTSLIM**

Reset Type	Reset Value	Note
LVD Reset	0CD8 06D6 <sub>H</sub>	
Cold PORST	0CD8 06D6 <sub>H</sub>	

## Power Management System for Low-End (PMSLE)

### 12.3.2.4 Standby and Wake-up Control Registers

#### Standby and Wake-up Control Register 0

##### PMSWCRO

##### Standby and Wake-up Control Register 0

(00B4<sub>H</sub>)

LVD Reset Value: 0010 02D0<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WUTW KEN	PORS TWKE N	SCRW KEN	PWRW KEN	PINBW KEN	PINAW KEN	ESR1 WKEN	ESR0 WKEN		BLNKFIL		0		STBYRAMSEL		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	r		rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PINBEDCON	PINBD FEN	PINAEDCON	PINAD FEN	ESR1EDCON	ESR1D FEN	ESR0EDCON	ESR0D FEN	VDDST BYEN	VEXTS TBYEN		0				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw			r

Field	Bits	Type	Description
VEXTSTBYEN	2	rw	<b>Standby Entry on VEXT Supply ramp-down</b> This bit field enables Standby Entry on VEXT supply ramp-down. This is supported only in case Standby domain is supplied separately via VEVRSSB supply pin and VEXT rail is switched off during Standby. The voltage threshold for entry is configured in EVRUVMON register. Current configuration is reflected in PMSWSTAT2.VEXTSTBYEN register bit. 0 <sub>B</sub> Standby Entry on VEXT supply ramp-down is disabled. 1 <sub>B</sub> Standby Entry triggered on a VEXT Supply undervoltage event (SWDUV). Blanking filter active on Standby mode entry.
VDDSTBYEN	3	rw	<b>Standby Entry on VDD Supply ramp-down</b> This bit field enables Standby Entry on VDD supply ramp-down. This is supported only in case Standby domain is supplied separately via VEVRSSB supply pin and VDD rail is switched off during Standby. The voltage threshold for entry is configured in EVRUVMON register. Current configuration is reflected in PMSWSTAT2.VDDSTBYEN register bit. 0 <sub>B</sub> Standby Entry on VDD supply ramp-down is disabled. 1 <sub>B</sub> Standby Entry triggered on a VDD Supply undervoltage event (VDDUV). Blanking filter active on Standby mode entry.
ESRODFEN	4	rw	<b>ESRO Digital Filter Enable</b> This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values) is enabled during normal RUN mode, then pulses less than 30ns are suppressed and pulses longer than 100ns will always result in a trigger. If the back-up clock is disabled in Standby mode and filter is running on 70 KHz Standby clock, then pulses less than 5 us are suppressed and pulses longer than 50 us will always result in a trigger. 0 <sub>B</sub> The filter is bypassed 1 <sub>B</sub> The filter is used

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>ESR0EDCON</b>	6:5	rw	<p><b>ESR0 Edge Detection Control</b></p> <p>This bit field defines the edge of a ESR0 wake-up trigger</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No trigger is generated</li> <li>01<sub>B</sub> A trigger is generated upon a rising edge</li> <li>10<sub>B</sub> A trigger is generated upon a falling edge</li> <li>11<sub>B</sub> A trigger is generated upon a rising OR falling edge</li> </ul>
<b>ESR1DFEN</b>	7	rw	<p><b>ESR1 Digital Filter Enable</b></p> <p>This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values) is enabled during normal RUN mode, then pulses less than 30ns are suppressed and pulses longer than 100ns will always result in a trigger. If the back-up clock is disabled in Standby mode and filter is running on 70 KHz Standby clock, then pulses less than 5 us are suppressed and pulses longer than 50 us will always result in a trigger.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> The filter is bypassed</li> <li>1<sub>B</sub> The filter is used</li> </ul>
<b>ESR1EDCON</b>	9:8	rw	<p><b>ESR1 Edge Detection Control</b></p> <p>This bit field defines the edge of a ESR1 wake-up trigger</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No trigger is generated</li> <li>01<sub>B</sub> A trigger is generated upon a rising edge</li> <li>10<sub>B</sub> A trigger is generated upon a falling edge</li> <li>11<sub>B</sub> A trigger is generated upon a rising OR falling edge</li> </ul>
<b>PINADFEN</b>	10	rw	<p><b>PINA Digital Filter Enable</b></p> <p>This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values) is enabled during normal RUN mode, then pulses less than 40ns are suppressed and pulses longer than 100ns will always result in a trigger. If the back-up clock is disabled in Standby mode and filter is running on 70 KHz Standby clock, then pulses less than 5 us are suppressed and pulses longer than 50 us will always result in a trigger.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> The filter is bypassed</li> <li>1<sub>B</sub> The filter is used</li> </ul>
<b>PINAEDCON</b>	12:11	rw	<p><b>PINA Edge Detection Control</b></p> <p>This bit field defines the edge of a Pin A wake-up trigger</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No trigger is generated</li> <li>01<sub>B</sub> A trigger is generated upon a rising edge</li> <li>10<sub>B</sub> A trigger is generated upon a falling edge</li> <li>11<sub>B</sub> A trigger is generated upon a rising OR falling edge</li> </ul>
<b>PINBDFEN</b>	13	rw	<p><b>PINB Digital Filter Enable</b></p> <p>This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values) is enabled during normal RUN mode, then pulses less than 40ns are suppressed and pulses longer than 100ns will always result in a trigger. If the back-up clock is disabled in Standby mode and filter is running on 70 KHz Standby clock, then pulses less than 5 us are suppressed and pulses longer than 50 us will always result in a trigger.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> The filter is bypassed</li> <li>1<sub>B</sub> The filter is used</li> </ul>

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description																												
<b>PINBEDCON</b>	15:14	rw	<p><b>PINB Edge Detection Control</b></p> <p>This bit field defines the edge of a Pin B wake-up trigger</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> No trigger is generated</li> <li>01<sub>B</sub> A trigger is generated upon a rising edge</li> <li>10<sub>B</sub> A trigger is generated upon a falling edge</li> <li>11<sub>B</sub> A trigger is generated upon a rising OR falling edge</li> </ul>																												
<b>STBYRAMSEL</b>	18:16	rw	<p><b>Standby RAM supply in Standby Mode</b></p> <p>This bit field configures the Standby RAM blocks to be kept supplied during Standby Mode from VDDPD supply rail. The current configuration is reflected in PMSWSTAT2.STBYRAM bitfield.</p> <p><i>Note:</i> All other bit combinations are reserved.</p> <ul style="list-style-type: none"> <li>000<sub>B</sub> Standby RAM is not supplied.</li> <li>001<sub>B</sub> Standby RAM (CPU0 dLMU RAM Lower Half) is supplied.</li> <li>010<sub>B</sub> Standby RAM (CPU0 dLMU RAM) is supplied.</li> <li>100<sub>B</sub> Reserved.</li> <li>111<sub>B</sub> Reserved.</li> </ul>																												
<b>BLNKFIL</b>	23:20	rw	<p><b>Blanking Filter delay for Wake-up</b></p> <p>This bitfield enables a nominal blanking filter delay time immediately after Standby entry only after which a valid wake-up event is recognized and reacted upon. The actual delay may vary +- 30% to this nominal value. Current configuration is reflected in PMSWSTAT2.BLNKFIL bitfield.</p> <p><i>Note:</i> All other bit combinations are reserved. Incase WUT is used as a wake-up source, the blanking filter should be configured for a period greater than 3x 70kHz clock cycles.</p> <table> <tbody> <tr><td>0<sub>H</sub></td><td>0 ms</td></tr> <tr><td>1<sub>H</sub></td><td>2,5 ms</td></tr> <tr><td>2<sub>H</sub></td><td>5 ms</td></tr> <tr><td>3<sub>H</sub></td><td>10 ms</td></tr> <tr><td>4<sub>H</sub></td><td>20 ms</td></tr> <tr><td>5<sub>H</sub></td><td>40 ms</td></tr> <tr><td>6<sub>H</sub></td><td>80 ms</td></tr> <tr><td>7<sub>H</sub></td><td>160 ms</td></tr> <tr><td>8<sub>H</sub></td><td>320 ms</td></tr> <tr><td>9<sub>H</sub></td><td>640 ms</td></tr> <tr><td>A<sub>H</sub></td><td>1280 ms</td></tr> <tr><td>B<sub>H</sub></td><td>2560 ms</td></tr> <tr><td>C<sub>H</sub></td><td>5120 ms</td></tr> <tr><td>D<sub>H</sub></td><td>10240 ms</td></tr> </tbody> </table>	0 <sub>H</sub>	0 ms	1 <sub>H</sub>	2,5 ms	2 <sub>H</sub>	5 ms	3 <sub>H</sub>	10 ms	4 <sub>H</sub>	20 ms	5 <sub>H</sub>	40 ms	6 <sub>H</sub>	80 ms	7 <sub>H</sub>	160 ms	8 <sub>H</sub>	320 ms	9 <sub>H</sub>	640 ms	A <sub>H</sub>	1280 ms	B <sub>H</sub>	2560 ms	C <sub>H</sub>	5120 ms	D <sub>H</sub>	10240 ms
0 <sub>H</sub>	0 ms																														
1 <sub>H</sub>	2,5 ms																														
2 <sub>H</sub>	5 ms																														
3 <sub>H</sub>	10 ms																														
4 <sub>H</sub>	20 ms																														
5 <sub>H</sub>	40 ms																														
6 <sub>H</sub>	80 ms																														
7 <sub>H</sub>	160 ms																														
8 <sub>H</sub>	320 ms																														
9 <sub>H</sub>	640 ms																														
A <sub>H</sub>	1280 ms																														
B <sub>H</sub>	2560 ms																														
C <sub>H</sub>	5120 ms																														
D <sub>H</sub>	10240 ms																														
<b>ESROWKEN</b>	24	rw	<p><b>ESR0 Wake-up enable from Standby</b></p> <p>This bit configures wake-up via ESR0 pin from STANDBY mode and current configuration is reflected in PMSWSTAT2.ESR0WKEN register bit.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> System wake-up via ESR0 pin is disabled.</li> <li>1<sub>B</sub> System wake-up is enabled via ESR0 pin.</li> </ul>																												

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>ESR1WKEN</b>	25	rw	<b>ESR1 Wake-up enable from Standby</b> This bit configures wake-up via ESR1 pin from STANDBY mode and current configuration is reflected in PMSWSTAT2.ESR1WKEN register bit. $0_B$ System wake-up via ESR1 pin is disabled. $1_B$ System wake-up is enabled via ESR1 pin.
<b>PINAWKEN</b>	26	rw	<b>Pin A Wake-up enable from Standby</b> This bit configures wake-up via PINA pin from STANDBY mode and current configuration is reflected in PMSWSTAT2.PINAWKEN register bit. $0_B$ System wake-up via Pin A is disabled. $1_B$ System wake-up is enabled via Pin A.
<b>PINBWKEN</b>	27	rw	<b>Pin B Wake-up enable from Standby</b> This bit configures wake-up via PINB pin from STANDBY mode and current configuration is reflected in PMSWSTAT2.PINBWKEN register bit. $0_B$ System wake-up via Pin B is disabled. $1_B$ System wake-up is enabled via Pin B.
<b>PWRWKEN</b>	28	rw	<b>Standby Wake-up Enable on VEXT Supply ramp-up</b> This bit field enables wake-up on VEXT supply ramp-up after blanking filter time has expired. This is supported only in case Standby domain is supplied separately via VEVRSB supply pin and VEXT rail is switched off during Standby. Current configuration is reflected in PMSWSTAT2.PWRWKEN register bit. $0_B$ Wake-up on VEXT supply ramp-down is disabled. Blanking filter configuration has no effect. $1_B$ Wake-up from standby on VEXT supply ramp-up is enabled after blanking filter time expiry.
<b>SCRWKEN</b>	29	rw	<b>Standby Controller Wake-up enable from Standby</b> This bit configures wake-up via SCR from STANDBY mode and current configuration is reflected in PMSWSTAT2.SCRWKEN register bit. $0_B$ System wake-up via 8 bit Standby Controller is disabled. $1_B$ System wake-up is enabled via 8 bit Standby Controller.
<b>PORSTWKEN</b>	30	rw	<b>PORST pin Wake-up enable from Standby</b> This bit configures wake-up via PORST pin from STANDBY mode and current configuration is reflected in PMSWSTAT2.PORSTWKEN register bit. $0_B$ System wake-up via PORST pin is disabled. $1_B$ System wake-up via PORST pin is enabled.
<b>WUTWKEN</b>	31	rw	<b>WUT Wake-up enable from Standby</b> This bit configures wake-up via WUT from STANDBY mode and current configuration is reflected in PMSWSTAT2.WUTWKEN register bit. $0_B$ System wake-up via Wake-up Timer is disabled. $1_B$ System wake-up is enabled via Wake-up Timer.
<b>0</b>	1:0, 19	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System for Low-End (PMSLE)

### Standby and Wake-up Control Register 2

#### PMSWCR2

#### Standby and Wake-up Control Register 2

(00B8<sub>H</sub>)LVD Reset Value: 0400 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
					<b>RST</b>	<b>SMUR ST</b>	<b>TCINT REQ</b>								<b>TCINT</b>
r				rh	rh	rh	rwh				rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				<b>SCRRS T</b>	<b>SCRW DT</b>	<b>SCREC C</b>	<b>0</b>								<b>SCRINT</b>
r				rwh	rwh	rwh	r				rh				

Field	Bits	Type	Description
<b>SCRINT</b>	7:0	rh	<b>Data exchange from Standby Controller to PMS main domain.</b> This bit field allows fast data exchange from SCR to PMS/CPUx. The data maybe read by CPUx consequent to an interrupt from the SCR to decode the interrupt. Incase SCR is enabled, at the end of the SCR Firmware routine, a value of 80H is set in SCRINT register to indicate that SCR has finished executing the startup code.
<b>SCRECC</b>	9	rwh	<b>SCR RAM ECC error / reset flag</b>  Note: <i>The flag is set by SCR and cleared by explicit write to the register bit. The flag is not cleared by SCR. While the SCR is being reset triggered by SCR RAM ECC error, this flag is set and clearing the flag is not possible for that duration.</i>  0 <sub>B</sub> No ECC error / reset reported by SCR. 1 <sub>B</sub> ECC error / reset was detected in SCR RAM.
<b>SCRWDT</b>	10	rwh	<b>SCR Watchdog Timer error / reset flag</b>  Note: <i>The flag is set by SCR and cleared by explicit write to the register bit. The flag is not cleared by SCR. While the SCR is being reset triggered by SCR watchdog, this flag is set and clearing the flag is not possible for that duration.</i>  0 <sub>B</sub> No WDT error / reset reported by SCR. 1 <sub>B</sub> WDT timer error / reset reported by SCR.
<b>SCRRST</b>	11	rwh	<b>SCR Software reset flag</b>  Note: <i>The flag is set by SCR and cleared by explicit write to the register bit. The flag is not cleared by SCR. While the SCR is being reset triggered by SCR software, this flag is set and clearing the flag is not possible for that duration.</i>  0 <sub>B</sub> No reset occurred in SCR. 1 <sub>B</sub> A reset has occurred in SCR.

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
TCINT	23:16	rw	<b>Data exchange from PMS main domain to Standby Controller.</b> This bit field allows fast data exchange from PMS to SCR. The data may be read by SCR consequent to an interrupt request (TCINTREQ) from PMS/CPUx to SCR to decode the interrupt.
TCINTREQ	24	rwh	<b>SW Interrupt request from PMS to Standby Controller.</b> Setting this bit triggers an interrupt to the 8 bit Standby controller.
SMURST	25	rh	<b>SMU Reset indication flag</b> $0_B$ No reset was issued by SMU. $1_B$ SMU issued an application or system reset.
RST	26	rh	<b>Application or System Reset indication flag</b> $0_B$ No application or system reset occurred. $1_B$ An application or system reset has occurred.
0	8, 15:12, 31:27	r	<b>Reserved</b> Read as 0; should be written with 0.

## Standby and Wake-up Control Register 3

### PMSWCR3

#### Standby and Wake-up Control Register 3

(00C0<sub>H</sub>)

LVD Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	WUTM ODE	WUTDI V	BUSY	WUTE N	0										
r	rw	rw	rh	rw	r										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WUTREL															
rw															

Field	Bits	Type	Description
WUTREL	23:0	rw	<b>WUT reload value.</b> The counter starts counting down from WUTREL value. The current value of counter is indicated in WUTCNT. On WUTCNT underflow, a reload WUTCNT = WUTREL takes place in auto reload mode.
WUTEN	27	rw	<b>WUT enable</b> This bit enables the Wake-up Timer. The status bit PMSWSTAT.WUTEN is set once Wake-up Timer is enabled. $0_B$ Wake-up timer (WUT) disable request $1_B$ Wake-up timer (WUT) enable request.

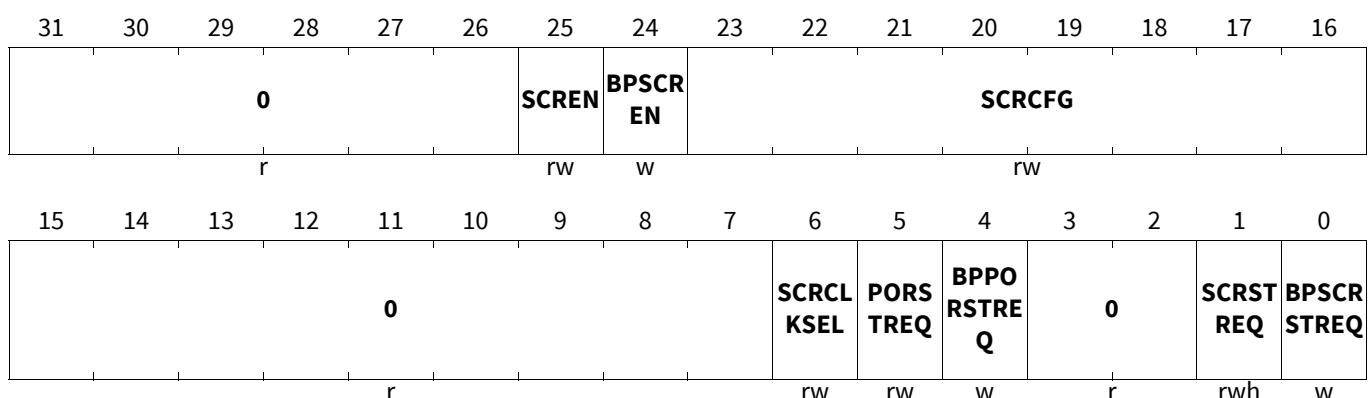
## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>BUSY</b>	28	rh	<b>Lock Status - LCK</b> This bit indicates that the register is busy owing to ongoing bus access. The register can be updated with a new value when BUSY bit is cleared. The register requires synchronization to the 70kHz clock domain on a register update. 0 <sub>B</sub> The register can be updated. 1 <sub>B</sub> The register update is ongoing. A write action may stall bus access for the time duration BUSY bit is set.
<b>WUTDIV</b>	29	rw	<b>WUT clock divider</b> A write to this register bitfield may trigger immediate update irrespective of the status of BUSY bit. 0 <sub>B</sub> Wake-up timer (WUT) clock = fSB = 70 KHz clock. 1 <sub>B</sub> Wake-up timer (WUT) clock = fSB (70 KHz) / 210.
<b>WUTMODE</b>	30	rw	<b>WUT mode selection</b> This bit configures the Wake-up Timer mode. The status bit PMSWSTAT.WUTMODE is respectively updated. A write to this register bitfield may trigger immediate update irrespective of the status of BUSY bit. 0 <sub>B</sub> Wake-up timer (WUT) auto reload mode selected 1 <sub>B</sub> Wake-up timer (WUT) auto stop mode selected.
<b>0</b>	26:24, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

### Standby and Wake-up Control Register 4

#### PMSWCR4

#### Standby and Wake-up Control Register 4

(00C4<sub>H</sub>)LVD Reset Value: 0000 0020<sub>H</sub>

Field	Bits	Type	Description
<b>BPSCRSTREQ</b>	0	w	<b>Standby Controller Reset request enable - SCRSTEN</b> 0 <sub>B</sub> Bit SCRSTREQ is not updated 1 <sub>B</sub> Bit SCRSTREQ can be updated

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>SCRSTREQ</b>	1	rwh	<p><b>Standby Controller Reset request</b></p> <p>0<sub>B</sub> No request for main reset of the 8 bit Standby Controller. (evr_scr_rst_req_i)</p> <p>1<sub>B</sub> 8 bit Standby Controller reset request.</p>
<b>BPPORSTREQ</b>	4	w	<p><b>Bit Protection for PORSTREQ - PORSTEN</b></p> <p>0<sub>B</sub> Bit PORSTREQ is not updated</p> <p>1<sub>B</sub> Bit PORSTREQ can be updated</p>
<b>PORSTREQ</b>	5	rw	<p><b>SCR Reset behavior on warm PORST in Normal RUN / SLEEP mode</b></p> <p>0<sub>B</sub> 8 bit Standby Controller is not reset when warm PORST pin is asserted.</p> <p>1<sub>B</sub> 8 bit Standby Controller is reset when warm PORST pin is asserted. warm PORST usage in normal and standby mode.</p>
<b>SCRCLKSEL</b>	6	rw	<p><b>Default Clock selection on Standby Mode Entry</b></p> <p>0<sub>B</sub> 100MHz oscillator can be enabled or disabled based on request from Standby Controller. By default 100 MHz Oscillator is requested by SCR in Standby Mode.</p> <p>1<sub>B</sub> 100MHz oscillator is always active irrespective of SCR requests. Thus both 70 KHz Oscillator and 100 MHz oscillator are active in Standby Mode.</p>
<b>SCRCFG</b>	23:16	rw	<p><b>Hardware configuration of the 8 bit SCR controller.</b></p> <p><b>Note:</b> Any change in SCRCFG is followed by a SCRSTREQ reset request of the 8 bit controller to start off in the chosen mode. All other bit combinations are reserved. Writing to PMSWCR4.SCRCFG with values != USERMODE1/0 will have an immediate effect on the enabling of debug pins.</p> <p>00<sub>H</sub> 8 bit XRAM is not programmed (default)      01<sub>H</sub> User Mode (Execution from 0000<sub>H</sub> XRAM address)      02<sub>H</sub> OCDS Mode (SCR DAP0_0/DAP1_0 pin mode)      03<sub>H</sub> OCDS Mode (SCR DAP0_1/DAP1_1 pin mode)      04<sub>H</sub> OCDS Mode (SCR SPD_0 pin mode)      05<sub>H</sub> OCDS Mode (SCR SPD_0 pin mode)      06<sub>H</sub> OCDS Mode (SCR SPD_1 pin mode)      07<sub>H</sub> OCDS Mode (SCR SPD_1 pin mode)      0A<sub>H</sub> OCDS Mode (SOC DAP mode)      0B<sub>H</sub> OCDS Mode (SOC DAP mode)      0C<sub>H</sub> OCDS Mode (SOC SPD mode)      0F<sub>H</sub> OCDS Mode (SOC SPD mode)</p>
<b>BPSCREN</b>	24	w	<p><b>Standby Controller Reset request enable</b></p> <p>0<sub>B</sub> Bit SCREN is not updated      1<sub>B</sub> Bit SCREN can be updated</p>
<b>SCREN</b>	25	rw	<p><b>Standby Controller Enable request</b></p> <p>SCR MBIST maybe activated independent of this bit.</p> <p>0<sub>B</sub> 8 bit Standby Controller is disabled      1<sub>B</sub> 8 bit Standby Controller is enabled</p>

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
0	3:2, 15:7, 31:26	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 466 Access Mode Restrictions of PMSWCR4 sorted by descending priority**

Mode Name	Access Mode		Description
write 1 to <b>BPSCRSTREQ</b>	rwh SCRSTREQ		
write 1 to <b>BPPORSTREQ</b>	rw PORSTREQ		
write 1 to <b>BPSCREN</b>	rw SCREN		
(default)	r PORSTREQ, SCREN		
	rh SCRSTREQ		

### Standby and Wake-up Control Register 5

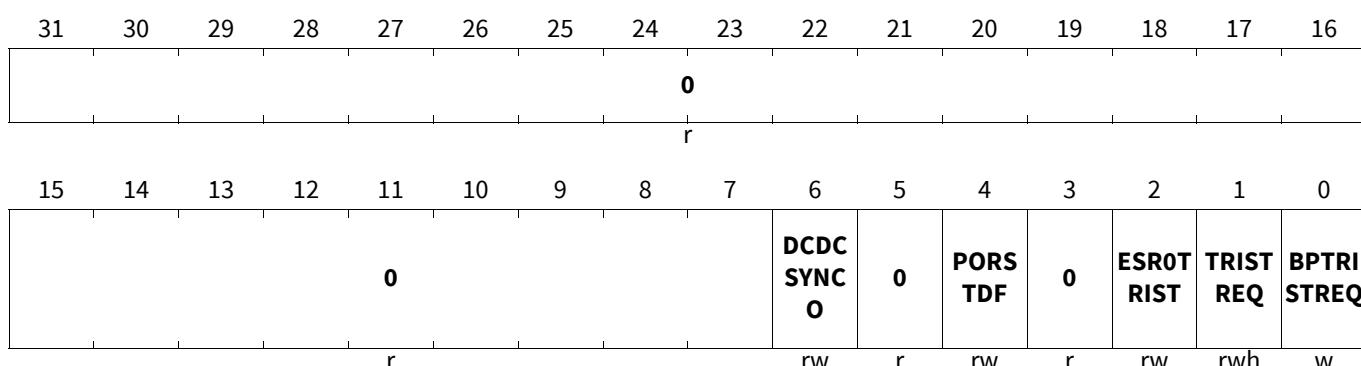
Additional PORST digital filter activated via PORSTDF bit provides additional spike filtering of at least tPORSTDF duration to provide enhanced immunity against spurious spikes. This is in addition to the inherent analog PORST filter delay of the PORST pad / pin as documented in the datasheet. After cold PORST this delay is by default inactive.

#### PMSWCR5

##### Standby and Wake-up Control Register 5

(00C8<sub>H</sub>)

LVD Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>BPTRISTREQ</b>	0	w	<b>Bit protection for Tristate request bit (TRISTREQ)</b> Setting this bit enables that bit TRISTREQ can be changed by a write operation. 0 <sub>B</sub> TRISTREQ keeps the previous state and cannot be changed. 1 <sub>B</sub> TRISTREQ bit can be changed with a write operation.

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>TRISTREQ</b>	1	rwh	<p><b>Tristate enable</b></p> <p>This bit decides whether pads behave as inputs with weak pull-up or tristate on reset assertion/de-assertion or Standby-Wake-up transition. After supply ramp-up or LVD reset, TRISTREQ = ! HWCFG6.</p> <p>0<sub>B</sub> No request to switch the input pad state of all the pads to tristate from pull-up (default reset state)</p> <p>1<sub>B</sub> Pad domain in tristate.</p>
<b>ESR0TRIST</b>	2	rw	<p><b>ESR0 Tristate enable</b></p> <p>This bit configures ESR0 pin behavior either as reset output or tristate during Standby mode if VEXT is supplied.</p> <p>0<sub>B</sub> ESR0 configured as reset output and is held low during Standby state (default reset state)</p> <p>1<sub>B</sub> ESR0 in tristate during Standby state.</p>
<b>PORSTDF</b>	4	rw	<p><b>PORST Digital Filter enable</b></p> <p>This bit field enables additional PORST digital filter ( tPORSTDF parameter ) to provide enhanced immunity against spurious spikes.</p> <p>0<sub>B</sub> PORST recognition delay = Analog PORST pad filter delay (default reset state).</p> <p>1<sub>B</sub> PORST recognition delay = Analog PORST pad filter delay + Digital filter delay.</p>
<b>DCDCSYNCO</b>	6	rw	<p><b>DC-DC Synchronisation Output Enable</b></p> <p>This bitfield enables the synchronisation output to synchronize the external SMPS regulator with respect to the internal EVRC regulator.</p> <p>0<sub>B</sub> DC-DC Synchronisation signal not available.</p> <p>1<sub>B</sub> DC-DC Synchronisation signal available.</p>
<b>0</b>	3, 5, 31:7	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 467 Access Mode Restrictions of PMSWCR5 sorted by descending priority**

Mode Name	Access Mode		Description
write 1 to <b>BPTRISTREQ</b>	rwh	TRISTREQ	
(default)	rh	TRISTREQ	

## Power Management System for Low-End (PMSLE)

### Standby WUT Counter Register

#### PMSWUTCNT

##### Standby WUT Counter Register

(00DC<sub>H</sub>)

LVD Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WUTCNT							
r															rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WUTCNT								rh							

Field	Bits	Type	Description
WUTCNT	23:0	rh	<b>WUT counter value.</b> The current WUT counter value is indicated in this register bitfield. The WUTCNT value may have a deviation of 3 additional clock cycles to the expected counter value owing to synchronization overheads. The WUT clock is based on standby 70 kHz clock with ~ + - 30% variation. The counter depending on the mode can run through a RUN to STANDBY to RUN mode transition without interruption.
0	31:24	r	<b>Reserved</b> Read as 0; should be written with 0.

### Standby and Wake-up Status Register

#### PMSWSTAT

##### Standby and Wake-up Status Register

(00D4<sub>H</sub>)

LVD Reset Value: 000A 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PINBI NT	PINAI NT	ESR1I NT	ESROI NT	0	WUTM ODE	WUTR UN	WUTE N	0				PORS TREQ	SCRCL K	SCRST	SCR
rh	rh	rh	rh	r	rh	rh	rh	r				rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				PORS TDF	0		ESROT RIST	TESTM ODE	TRIST	HWCF G5	HWCF G4	0	HWCFGEV	0	
r				rh	r		rh	rh	rh	rh	rh	r	rh	rh	r

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>HWCFGEV</b>	2:1	rh	<p><b>EVR Hardware Configuration status</b></p> <p>This bit field indicates the supply configuration latched by the EVR from HWCFG[2:1] during a cold startup based on which EVRx regulators are consequently started. The latched configuration is used during STANDBY-RUN transition to reselect EVR mode.</p> <p>00<sub>B</sub> EVRC inactive, EVR33 inactive. 01<sub>B</sub> EVRC inactive, EVR33 active. 10<sub>B</sub> EVRC active, EVR33 inactive. 11<sub>B</sub> EVRC active, EVR33 active.</p>
<b>HWCFG4</b>	4	rh	<p><b>Hardware Configuration Pin 4 status</b></p> <p>This bit field indicates the latched level of HWCFG[4] during a cold startup.</p>
<b>HWCFG5</b>	5	rh	<p><b>Hardware Configuration Pin 5 status</b></p> <p>This bit field indicates the latched level of HWCFG[5] during a cold startup.</p>
<b>TRIST</b>	6	rh	<p><b>Pad Tristate / Pull-up status</b></p> <p>This bit indicates whether pads are configured as inputs with weak pull-up or as tristate during/after reset or after wake-up. At start-up, the value latched from HWCFG[6] pin decides the default state and is reflected in TRIST status bit. This bit may be later updated when PMSWCR5.TRISTREQ is set to override initial latched status from HWCFG[6].</p> <p>0<sub>B</sub> Pads configured as inputs with weak pull-up. 1<sub>B</sub> Pads are in tristate.</p>
<b>TESTMODE</b>	7	rh	<p><b>TESTMODE Pin status</b></p> <p>This bit field indicates the latched level of TESTMODE pin during a cold startup.</p>
<b>ESR0TRIST</b>	8	rh	<p><b>ESR0 pin status during Standby</b></p> <p>This bit indicates if ESR0 pin is configured as reset output or tristate during Standby mode &amp; transitions if VEXT is supplied. This bit is updated when PMSWCR5.ESR0TRIST is set.</p> <p>0<sub>B</sub> ESR0 configured as reset output and is held low during Standby state (default reset state) 1<sub>B</sub> ESR0 in tristate during Standby state.</p>
<b>PORSTDF</b>	11	rh	<p><b>PORST Digital Filter status</b></p> <p>This bit field indicates whether additional PORST digital filter is activated. This bit is updated when PMSWCR5.PORSTDF is set.</p> <p>0<sub>B</sub> PORST recognition delay = Analog PORST pad filter delay (default reset state). 1<sub>B</sub> PORST recognition delay = Analog PORST pad filter delay + Digital filter delay.</p>

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>SCR</b>	16	rh	<p><b>Standby Controller status</b></p> <p>This bit indicates whether SCR is enabled. This bit is updated when PMSWCR4.SCREN bit is set.</p> <p>0<sub>B</sub> 8 bit Standby Controller is disabled 1<sub>B</sub> 8 bit Standby Controller is enabled</p>
<b>SCRST</b>	17	rh	<p><b>Standby Controller Reset Indication flag</b></p> <p>This bit is set after a power-on reset as SCR is in reset state. This bit is consequently set when a reset is issued via PMSWCR4.SCRSTREQ bit. This status flag is set on every SCR reset caused by any SCR reset source.</p> <p>0<sub>B</sub> No reset of Standby controller took place. 1<sub>B</sub> Reset of Standby controller took place. (evr_scr_rst_o)</p>
<b>SCRCLK</b>	18	rh	<p><b>Current Clock configuration for SCR before Standby Mode Entry</b></p> <p>This bit is updated when PMSWCR4.SCRCLKSEL bit is set.</p> <p>0<sub>B</sub> Only 70 KHz Oscillator is active in Standby Mode. 1<sub>B</sub> Both 70 KHz Oscillator and 100 MHz oscillator are active in Standby Mode.</p>
<b>PORSTREQ</b>	19	rh	<p><b>Standby Controller Reset on warm PORST</b></p> <p>This bit is updated when PMSWCR4.PORSTREQ bit is set.</p> <p>0<sub>B</sub> 8 bit Standby Controller clock is not reset when warm PORST pin is asserted. 1<sub>B</sub> 8 bit Standby Controller is reset when warm PORST pin is asserted.</p>
<b>WUTEN</b>	24	rh	<p><b>WUT Enable status</b></p> <p>This bit indicates whether WUT is enabled. This bit is updated when PMSWCR3.WUTEN bit is updated.</p> <p>0<sub>B</sub> Wake-up timer (WUT) is disabled. 1<sub>B</sub> Wake-up timer (WUT) is enabled.</p>
<b>WUTRUN</b>	25	rh	<p><b>WUT Run status</b></p> <p>This bit indicates whether WUT is currently running. Due to synchronization to 70 KHz ( fSB ) WUT clock, setting of flag after enable may take up to 55 us.</p> <p>0<sub>B</sub> Wake-up timer (WUT) is inactive. 1<sub>B</sub> Wake-up timer (WUT) is active.</p>
<b>WUTMODE</b>	26	rh	<p><b>WUT Mode status</b></p> <p>This bit indicates the current WUT mode. This bit is updated when PMSWCR3.WUTMODE bit is updated.</p> <p>0<sub>B</sub> Wake-up timer (WUT) auto reload mode is selected 1<sub>B</sub> Wake-up timer (WUT) auto stop mode is selected.</p>
<b>ESR0INT</b>	28	rh	<p><b>ESR0 Interrupt flag</b></p> <p>In case interrupt was triggered by ESR0 pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via PMSWSTATCLR.ESR0INTCLR bit after interrupt is serviced.</p> <p>0<sub>B</sub> No interrupt event detected on ESR0 input. 1<sub>B</sub> An interrupt event as defined by PMSWCR0. ESR0EDCON detected on ESR0 input.</p>

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
ESR1INT	29	rh	<b>ESR1 Interrupt flag</b> In case interrupt was triggered by ESR1 pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via PMSWSTATCLR.ESR1INTCLR bit after interrupt is serviced. 0 <sub>B</sub> No interrupt event detected on ESR1 input. 1 <sub>B</sub> An interrupt event as defined by PMSWCR0. ESR1EDCON detected on ESR1 input.
PINAINT	30	rh	<b>Pin A Interrupt flag</b> In case interrupt was triggered by PINA pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via PMSWSTATCLR.PINAINTCLR bit after interrupt is serviced. 0 <sub>B</sub> No interrupt event detected on Pin A input. 1 <sub>B</sub> An interrupt event as defined by PMSWCR0. PINAEDCON detected on Pin A input.
PINBINT	31	rh	<b>Pin B Interrupt flag</b> In case interrupt was triggered by PINB pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via PMSWSTATCLR.PINBINTCLR bit after interrupt is serviced. 0 <sub>B</sub> No interrupt event detected on the Pin B input. 1 <sub>B</sub> An interrupt event as defined by PMSWCR0. PINBEDCON detected on Pin B input.
0	0, 3, 10:9, 15:12, 23:20, 27	r	<b>Reserved</b> Read as 0; should be written with 0.

## Standby and Wake-up Status Register 2

## PMSWSTAT2

## Standby and Wake-up Status Register 2

(00D8<sub>H</sub>)LVD Reset Value: 0010 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WUTW KEN	PORS TWKE N	SCRW KEN	PWRW KEN	PINBW KEN	PINAW KEN	ESR1 WKEN	ESR0 WKEN		BLNKFIL		VEXTS TBYEN		STBYRAM		
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WUTO VRUN	PORS TOVR UN	SCRO VRUN	VDDST BYEN	PINBO VRUN	PINAO VRUN	ESR10 VRUN	ESR00 VRUN	WUTW KP	PORS TWKP	SCRW KP	PWRW KP	PINBW KP	PINAW KP	ESR1 WKP	ESR0 WKP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>ESR0WKP</b>	0	rh	<b>ESR0 Wake-up flag</b> In case wake-up was triggered by ESR0 pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.ESR0WKPCR bit before next STANDBY entry. 0 <sub>B</sub> No wake-up event detected on ESR0 input during STANDBY. 1 <sub>B</sub> An event as defined by PMSWCR0. ESR0EDCON detected on ESR0 input.
<b>ESR1WKP</b>	1	rh	<b>ESR1 Wake-up flag</b> In case wake-up was triggered by ESR1 pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.ESR1WKPCR bit before next STANDBY entry. 0 <sub>B</sub> No wake-up event detected on ESR1 input during STANDBY. 1 <sub>B</sub> An event as defined by PMSWCR0. ESR1EDCON detected on ESR1 input.
<b>PINAWKP</b>	2	rh	<b>Pin Wake-up flag</b> In case wake-up was triggered by PINA pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PINAWKPCR bit before next STANDBY entry. 0 <sub>B</sub> No wake-up event detected on Pin A input during STANDBY. 1 <sub>B</sub> An event as defined by PMSWCR0. PINAEDCON detected on Pin A input.
<b>PINBWKP</b>	3	rh	<b>Pin B Wake-up flag</b> In case wake-up was triggered by PINB pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PINBKPCR bit before next STANDBY entry. 0 <sub>B</sub> No wake-up event occurred on the Pin B input during STANDBY. 1 <sub>B</sub> An event as defined by PMSWCR0. PINBEDCON detected on Pin B input.
<b>PWRWKP</b>	4	rh	<b>Wake-up event on VEXT Supply ramp-up</b> In case wake-up was triggered by VEXT ramp-up pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PWRWKPCR bit before next STANDBY entry. 0 <sub>B</sub> No VEXT supply wake-up event detected. 1 <sub>B</sub> VEXT Monitor threshold exceeded on VEXT supply ramp-up leading to System Wake-up from STANDBY.
<b>SCRWKP</b>	5	rh	<b>SCR Wake-up flag</b> In case wake-up is triggered by SCR to the main controller during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.SCRWKPCR bit before next STANDBY entry. 0 <sub>B</sub> No SCR wake-up event detected. 1 <sub>B</sub> A SCR wake-up event occurred.

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>PORSTWKP</b>	6	rh	<p><b>PORST Wake-up flag</b></p> <p>In case wake-up was triggered by PORST pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PORSTWKPCR bit before next STANDBY entry.</p> <p>0<sub>B</sub> No wake-up event detected on PORST input during STANDBY if enabled via PMSWCR0.PORSTWKEN bit.</p> <p>1<sub>B</sub> A wake-up event detected on PORST input if enabled via PMSWCR0.PORSTWKEN bit.</p>
<b>WUTWKP</b>	7	rh	<p><b>WUT Wake-up flag</b></p> <p>In case wake-up was triggered by Wake-up timer during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.WUTWKPCR bit before next STANDBY entry.</p> <p>0<sub>B</sub> No wake-up event detected due to WUT underflow.</p> <p>1<sub>B</sub> A wake-up event from STANDBY was detected due to WUT underflow.</p>
<b>ESR0OVRUN</b>	8	rh	<p><b>ESR0 Overrun status flag</b></p> <p>This flag indicates that a consecutive ESR0 wake-up event occurred while ESR0WKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.ESR0OVRUNCLR bit before next STANDBY entry.</p> <p>0<sub>B</sub> No overrun condition detected on ESR0 input.</p> <p>1<sub>B</sub> An overrun condition detected on ESR0 input.</p>
<b>ESR1OVRUN</b>	9	rh	<p><b>ESR1 Overrun status flag</b></p> <p>This flag indicates that a consecutive ESR1 wake-up event occurred while ESR1WKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.ESR1OVRUNCLR bit before next STANDBY entry.</p> <p>0<sub>B</sub> No overrun condition detected on ESR1 input.</p> <p>1<sub>B</sub> An overrun condition detected on ESR1 input.</p>
<b>PINAOVRUN</b>	10	rh	<p><b>Pin A Overrun status flag</b></p> <p>This flag indicates that a consecutive PINA wake-up event occurred while PINAWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PINAOVRUNCLR bit before next STANDBY entry.</p> <p>0<sub>B</sub> No overrun condition detected on Pin A input.</p> <p>1<sub>B</sub> An overrun condition detected on Pin A input.</p>
<b>PINBOVRUN</b>	11	rh	<p><b>Pin B Overrun status flag</b></p> <p>This flag indicates that a consecutive PINB wake-up event occurred while PINBWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PINBOVRUNCLR bit before next STANDBY entry.</p> <p>0<sub>B</sub> No overrun condition detected on Pin B input.</p> <p>1<sub>B</sub> An overrun condition detected on Pin B input.</p>

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>VDDSTBYEN</b>	12	rh	<p><b>Standby Entry Enable status on VDD Supply ramp-down - VDDSTBYWKEN</b></p> <p>This bit indicates that Standby Entry may be triggered on a VDD Supply undervoltage event (VDDUV). This is supported only when Standby domain is supplied separately by VEVRSB Standby supply pin. This bit is updated when PMSWCRO.VDDSTBYWKEN bit is updated.</p> <p>0<sub>B</sub> 0 Standby Entry on VDD supply ramp-down is disabled. 1<sub>B</sub> 1 Standby Entry is enabled on a VDD Supply undervoltage event (SWDUV). Blanking filter active on Standby mode entry.</p>
<b>SCROVRUN</b>	13	rh	<p><b>SCR Overrun status flag</b></p> <p>This flag indicates that a consecutive SCR wake-up event occurred while SCRWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.SCROVRUNCLR bit before next STANDBY entry.</p> <p>0<sub>B</sub> No overrun condition detected of SCR wake-up event. 1<sub>B</sub> An overrun condition detected of SCR wake-up event.</p>
<b>PORSTOVRUN</b>	14	rh	<p><b>PORST Overrun status flag</b></p> <p>This flag indicates that a consecutive PORST wake-up event occurred while PORSTWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.PORSTOVRUNCLR bit before next STANDBY entry.</p> <p>0<sub>B</sub> No overrun condition detected on PORST input if enabled via PMSWCRO.PORSTWKEN bit. 1<sub>B</sub> An overrun condition detected on PORST input if enabled via PMSWCRO.PORSTWKEN bit.</p>
<b>WUTOVRUN</b>	15	rh	<p><b>WUT Overrun status flag</b></p> <p>This flag indicates that a consecutive WUT wake-up event occurred while WUTWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via PMSWSTATCLR.WUTOVRUNCLR bit before next STANDBY entry. WUTREL need to be greater than 10 during Standby mode to be able to latch consecutive WUT underflow events and update the WUTOVRRUN register bitfield.</p> <p>0<sub>B</sub> No overrun condition detected of WUT events. 1<sub>B</sub> An overrun condition detected of WUT events.</p>

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description																												
<b>STBYRAM</b>	18:16	rh	<p><b>Standby RAM Supply status</b></p> <p>This bit field indicates whether Standby RAM was supplied during Standby Mode and to infer status after a wake-up event. This bit is updated when PMSWCR0.STBYRAMSEL is set.</p> <p><b>Note:</b> <i>All other bit combinations are reserved. In case of VDDPD Standby supply fail or VEVRSB supply fail leading to LVD reset (indicated also in RSTSTAT.STBYR), the STBYRAM status bit is reset to 000<sub>B</sub> to indicate that Standby RAM contents may be corrupted.</i></p> <p>000<sub>B</sub> Standby RAM is not supplied.      001<sub>B</sub> Standby RAM (CPU0 dLMU RAM Lower Half) is supplied.      010<sub>B</sub> Standby RAM (CPU0 dLMU RAM) is supplied.      100<sub>B</sub> Reserved.      111<sub>B</sub> Reserved.</p>																												
<b>VEXTSTBYEN</b>	19	rh	<p><b>Standby Entry Enable status on VEXT Supply ramp-down - VEXTSTBYWKEN</b></p> <p>This bit indicates that Standby Entry may be triggered on a VEXT Supply undervoltage event (SWDUV). This is supported only when Standby domain is supplied separately by VEVRSB Standby supply pin. This bit is updated when PMSWCR0.VEXTSTBYWKEN bit is updated.</p> <p>0<sub>B</sub> 0 Standby Entry on VEXT supply ramp-down is disabled.      1<sub>B</sub> 1 Standby Entry is enabled on a VEXT Supply undervoltage event (SWDUV). Blanking filter active on Standby mode entry.</p>																												
<b>BLNKFIL</b>	23:20	rh	<p><b>Blanking Filter Delay for VEXT Supply Wake-up</b></p> <p>This bit field indicates the Blanking filter configuration. This bit field is updated with the value configured in PMSWCR0.BLNKFIL bitfield.</p> <p><b>Note:</b> <i>All other bit combinations are reserved.</i></p> <table> <tbody> <tr> <td>0<sub>H</sub></td> <td>0 ms</td> </tr> <tr> <td>1<sub>H</sub></td> <td>2,5 ms</td> </tr> <tr> <td>2<sub>H</sub></td> <td>5 ms</td> </tr> <tr> <td>3<sub>H</sub></td> <td>10 ms</td> </tr> <tr> <td>4<sub>H</sub></td> <td>20 ms</td> </tr> <tr> <td>5<sub>H</sub></td> <td>40 ms</td> </tr> <tr> <td>6<sub>H</sub></td> <td>80 ms</td> </tr> <tr> <td>7<sub>H</sub></td> <td>160 ms</td> </tr> <tr> <td>8<sub>H</sub></td> <td>320 ms</td> </tr> <tr> <td>9<sub>H</sub></td> <td>640 ms</td> </tr> <tr> <td>A<sub>H</sub></td> <td>1280 ms</td> </tr> <tr> <td>B<sub>H</sub></td> <td>2560 ms</td> </tr> <tr> <td>C<sub>H</sub></td> <td>5120 ms</td> </tr> <tr> <td>D<sub>H</sub></td> <td>10240 ms</td> </tr> </tbody> </table>	0 <sub>H</sub>	0 ms	1 <sub>H</sub>	2,5 ms	2 <sub>H</sub>	5 ms	3 <sub>H</sub>	10 ms	4 <sub>H</sub>	20 ms	5 <sub>H</sub>	40 ms	6 <sub>H</sub>	80 ms	7 <sub>H</sub>	160 ms	8 <sub>H</sub>	320 ms	9 <sub>H</sub>	640 ms	A <sub>H</sub>	1280 ms	B <sub>H</sub>	2560 ms	C <sub>H</sub>	5120 ms	D <sub>H</sub>	10240 ms
0 <sub>H</sub>	0 ms																														
1 <sub>H</sub>	2,5 ms																														
2 <sub>H</sub>	5 ms																														
3 <sub>H</sub>	10 ms																														
4 <sub>H</sub>	20 ms																														
5 <sub>H</sub>	40 ms																														
6 <sub>H</sub>	80 ms																														
7 <sub>H</sub>	160 ms																														
8 <sub>H</sub>	320 ms																														
9 <sub>H</sub>	640 ms																														
A <sub>H</sub>	1280 ms																														
B <sub>H</sub>	2560 ms																														
C <sub>H</sub>	5120 ms																														
D <sub>H</sub>	10240 ms																														

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>ESR0WKEN</b>	24	rh	<b>ESR0 Wake-up enable status</b> This bit indicates that ESR0 is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.ESR0WKEN bit is updated. 0 <sub>B</sub> Wake-up from Standby via ESR0 is disabled. 1 <sub>B</sub> Wake-up from Standby via ESR0 is enabled.
<b>ESR1WKEN</b>	25	rh	<b>ESR1 Wake-up enable status</b> This bit indicates that ESR1 is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.ESR1WKEN bit is updated. 0 <sub>B</sub> Wake-up from Standby via ESR1 is disabled. 1 <sub>B</sub> Wake-up from Standby via ESR1 is enabled.
<b>PINAWKEN</b>	26	rh	<b>Pin A Wake-up enable status</b> This bit indicates that PINA is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.PINAWKEN bit is updated. 0 <sub>B</sub> Wake-up from Standby via PINA is disabled. 1 <sub>B</sub> Wake-up from Standby via PINA is enabled.
<b>PINBWKEN</b>	27	rh	<b>Pin B Wake-up enable status</b> This bit indicates that PINB is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.PINBWKEN bit is updated. 0 <sub>B</sub> Wake-up from Standby via PINB is disabled. 1 <sub>B</sub> Wake-up from Standby via PINB is enabled.
<b>PWRWKEN</b>	28	rh	<b>Standby Wake-up Enable status on VEXT Supply ramp-up</b> This bit indicates that VEXT detector is enabled to trigger wake-up from Standby during VEXT supply ramp-up after blanking filter time has expired. This is supported only when Standby domain is supplied separately by VEVRSB Standby supply pin. This bit is updated when PMSWCR0.PWRWKEN bit is updated. 0 <sub>B</sub> Wake-up on VEXT supply ramp-down disabled. Blanking filter configuration has no effect. 1 <sub>B</sub> Standby Wake-up on VEXT supply ramp-up is enabled after blanking filter expiry.
<b>SCRWKEN</b>	29	rh	<b>Standby Controller Wake-up Enable status</b> This bit indicates that SCR is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.SCRWKEN bit is updated. 0 <sub>B</sub> Wake-up from Standby via SCR is disabled. 1 <sub>B</sub> Wake-up from Standby via SCR is enabled.
<b>PORSTWKEN</b>	30	rh	<b>PORST pin Wake-up enable status from Standby</b> This bit indicates that wake-up via PORST pin is enabled during STANDBY mode. This bit is updated when PMSWCR0. PORSTWKEN bit is updated. 0 <sub>B</sub> System wake-up via PORST pin is disabled. 1 <sub>B</sub> System wake-up via PORST pin is enabled.
<b>WUTWKEN</b>	31	rh	<b>WUT Wake-up enable status</b> This bit indicates that WUT is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.WUTWKEN bit is updated. 0 <sub>B</sub> Wake-up from Standby via WUT is disabled. 1 <sub>B</sub> Wake-up from Standby via WUT is enabled.

## Power Management System for Low-End (PMSLE)

### Standby and Wake-up Status Clear Register

#### PMSWSTATCLR

**Standby and Wake-up Status Clear Register (00E8<sub>H</sub>)**

**LVD Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PINBI NTCLR	PINAI NTCLR	ESR1I NTCLR	ESROI NTCLR						0						SCRST CLR
W	W	W	W						r						W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WUTO VRUN CLR	PORS TOVR UNCL R	SCRO VRUN CLR	0	PINBO VRUN CLR	PINA0 VRUN CLR	ESR10 VRUN CLR	ESR00 VRUN CLR	WUTW KPCLR	PORS TWKP CLR	SCRW KPCLR	PWRW KPCLR	PINBW KPCLR	PINAW KPCLR	ESR1 WKPC LR	ESR0 WKPC LR
W	W	W	r	W	W	W	W	W	W	W	W	W	W	W	W

Field	Bits	Type	Description
ESR0WKPCLR	0	w	<b>ESR0 Wake-up indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.ESR0WKP bit cleared.
ESR1WKPCLR	1	w	<b>ESR1 Wake-up indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.ESR1WKP bit cleared.
PINAWKPCLR	2	w	<b>PINA Wake-up indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.PINAWKP bit cleared.
PINBWKPCLR	3	w	<b>PINB Wake-up indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.PINBWKP bit cleared.
PWRWKPCLR	4	w	<b>PWRWKP Wake-up indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.PWRWKP bit cleared.
SCRWKPCLR	5	w	<b>SCR Wake-up indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.SCRWKP bit cleared.
PORSTWKPCR	6	w	<b>PORST Wake-up indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.PORSTWKP bit cleared.
WUTWKPCLR	7	w	<b>WUT Wake-up indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.WUTWKP bit cleared.
ESR0OVRUNC LR	8	w	<b>ESR0 Overrun status indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.ESR0OVRUN bit cleared.

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>ESR1OVRUNC</b> <b>LR</b>	9	w	<b>ESR1 Overrun status indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.ESR1OVRUN bit cleared.
<b>PINAOVRUNC</b> <b>LR</b>	10	w	<b>PINA Overrun status indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.PINAOVRUN bit cleared.
<b>PINBOVRUNC</b> <b>LR</b>	11	w	<b>PINB Overrun status indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.PINBOVRUN bit cleared.
<b>SCROVRUNCL</b> <b>R</b>	13	w	<b>SCR Overrun status indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.SCROVRUN bit cleared.
<b>PORSTOVRUN</b> <b>CLR</b>	14	w	<b>PORST Overrun status indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.PORSTOVRUN bit cleared.
<b>WUTOVRUNC</b> <b>LR</b>	15	w	<b>WUT Overrun status indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT2.WUTOVRUN bit cleared.
<b>SCRSTCLR</b>	16	w	<b>Standby controller SCRST indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT.SCRST bit cleared.
<b>ESR0INTCLR</b>	28	w	<b>ESR0 Interrupt indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT.ESR0INT bit cleared.
<b>ESR1INTCLR</b>	29	w	<b>ESR1 Interrupt indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT.ESR1INT bit cleared.
<b>PINAINTCLR</b>	30	w	<b>PINA Interrupt indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT.PINAINT bit cleared.
<b>PINBINTCLR</b>	31	w	<b>PINB Interrupt indication flag clear</b> 0 <sub>B</sub> No action 1 <sub>B</sub> PMSWSTAT.PINBINT bit cleared.
<b>0</b>	12, 27:17	r	<b>Reserved</b> Read as 0; should be written with 0.

## 12.3.2.5 OCDS Trigger Bus Configuration Registers (OTGB)

Access are only supported for byte, half-word and word data and requires Supervisor Mode.

## Power Management System for Low-End (PMSLE)

### OCDS Trigger Set Select Register

#### OTSS

#### OCDS Trigger Set Select Register

(01E0<sub>H</sub>)

Reset Value: [Table 468](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
r									r						
0				OTGB1				0				OTGB0			

Field	Bits	Type	Description
OTGB0	3:0	rw	<b>Trigger Set for OTGB0</b> 0 <sub>H</sub> No Trigger Set selected 1 <sub>H</sub> Trigger Set TS16_ADCMON 2 <sub>H</sub> Trigger Set TS16_EVRC <b>others</b> , reserved
OTGB1	11:8	rw	<b>Trigger Set for OTGB1</b> 0 <sub>H</sub> No Trigger Set selected 1 <sub>H</sub> Trigger Set TS16_ADCMON 2 <sub>H</sub> Trigger Set TS16_EVRC <b>others</b> , reserved
0	7:4, 15:12, 31:16	r	<b>Reserved</b> Read as 0; must be written with 0.

**Table 468 Reset Values of OTSS**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Debug Reset	0000 0000 <sub>H</sub>	

### OCDS Trigger Set Control 0 Register

#### OTSC0

#### OCDS Trigger Set Control 0 Register

(01E4<sub>H</sub>)

Reset Value: [Table 469](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r									r						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
r									r						
0				B1HAM				0				B1LAM			
0				B0HAM				0				B0LAM			

## Power Management System for Low-End (PMSLE)

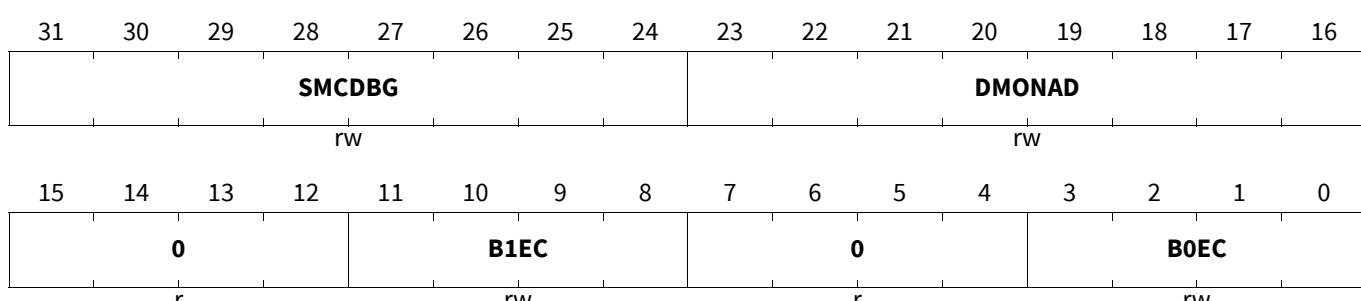
Field	Bits	Type	Description
<b>B0LAM</b>	3:0	rw	<b>OTGB0 TS16_ADCMON Low Byte</b> 0 <sub>H</sub> No Module selected 1 <sub>H</sub> PRADCCV 2 <sub>H</sub> PRADC33V 3 <sub>H</sub> PRADCSWDV 4 <sub>H</sub> PRADCFBCV 5 <sub>H</sub> SECADCCV 6 <sub>H</sub> SECADC33V 7 <sub>H</sub> SECADCSWDV 8 <sub>H</sub> SECADCPRE 9 <sub>H</sub> SECACDSB A <sub>H</sub> SECACVDDM B <sub>H</sub> DTSRESULTL C <sub>H</sub> DTSRESULTH <b>others</b> , reserved
<b>B0HAM</b>	11:8	rw	<b>OTGB0 TS16_ADCMON High Byte</b> 0 <sub>H</sub> No Module selected 1 <sub>H</sub> PRADCCV 2 <sub>H</sub> PRADC33V 3 <sub>H</sub> PRADCSWDV 4 <sub>H</sub> PRADCFBCV 5 <sub>H</sub> SECADCCV 6 <sub>H</sub> SECADC33V 7 <sub>H</sub> SECADCSWDV 8 <sub>H</sub> SECADCPRE 9 <sub>H</sub> SECACDSB A <sub>H</sub> SECACVDDM B <sub>H</sub> DTSRESULTL C <sub>H</sub> DTSRESULTH <b>others</b> , reserved
<b>B1LAM</b>	19:16	rw	<b>OTGB1 TS16_ADCMON Low Byte</b> 0 <sub>H</sub> No Module selected 1 <sub>H</sub> PRADCCV 2 <sub>H</sub> PRADC33V 3 <sub>H</sub> PRADCSWDV 4 <sub>H</sub> PRADCFBCV 5 <sub>H</sub> SECADCCV 6 <sub>H</sub> SECADC33V 7 <sub>H</sub> SECADCSWDV 8 <sub>H</sub> SECADCPRE 9 <sub>H</sub> SECACDSB A <sub>H</sub> SECACVDDM B <sub>H</sub> DTSRESULTL C <sub>H</sub> DTSRESULTH <b>others</b> , reserved

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>B1HAM</b>	27:24	rw	<b>OTGB1 TS16_ADCMON High Byte</b> 0 <sub>H</sub> No Module selected 1 <sub>H</sub> PRADCCV 2 <sub>H</sub> PRADC33V 3 <sub>H</sub> PRADCSDV 4 <sub>H</sub> PRADCFCV 5 <sub>H</sub> SECADCCV 6 <sub>H</sub> SECADC33V 7 <sub>H</sub> SECADCSDV 8 <sub>H</sub> SECADCPRE 9 <sub>H</sub> SECADCSB A <sub>H</sub> SECADCVDDM B <sub>H</sub> DTSRESULTL C <sub>H</sub> DTSRESULTH <b>others</b> , reserved
<b>0</b>	7:4, 15:12, 23:20, 31:28	r	<b>Reserved</b> Read as 0; must be written with 0.

**Table 469 Reset Values of OTSCO**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Debug Reset	0000 0000 <sub>H</sub>	

**OCDS Trigger Set Control 1 Register****OTSC1****OCDS Trigger Set Control 1 Register**(01E8<sub>H</sub>)**Reset Value: Table 470**

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
B0EC	3:0	rw	<b>OTGB0_TS16_EVRCON</b> 0 <sub>H</sub> No Module selected 1 <sub>H</sub> CONDUCTANCE 2 <sub>H</sub> EVRCOUT 3 <sub>H</sub> EVR33OUT 4 <sub>H</sub> WUTCNT 5 <sub>H</sub> TCSCRINT <b>others</b> , reserved
B1EC	11:8	rw	<b>OTGB1_TS16_EVRCON</b> 0 <sub>H</sub> No Module selected 1 <sub>H</sub> CONDUCTANCE 2 <sub>H</sub> EVRCOUT 3 <sub>H</sub> EVR33OUT 4 <sub>H</sub> WUTCNT 5 <sub>H</sub> TCSCRINT <b>others</b> , reserved
DMONAD	23:16	rw	<b>OTGB0_TS16_EVRCON DMONAD</b> The multiplexer signal selection documented in DMONAD coding table.
SMCDBG	31:24	rw	<b>OTGB0_TS16_EVRCON SMCDBG</b> Reserved for future extensions.
0	7:4, 15:12	r	<b>Reserved</b> Read as 0; must be written with 0.

**Table 470 Reset Values of OTSC1**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Debug Reset	0000 0000 <sub>H</sub>	

**Access Enable Register 0**

The Access Enable Register 0 restricts write access to all PMS registers so that they may only be written by specified bus masters (e.g. CPUs). See the Bus chapter for the mapping of TAG ID to specific system masters and CPUs).

**ACCENO****Access Enable Register 0** **(01F8<sub>H</sub>)** **Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

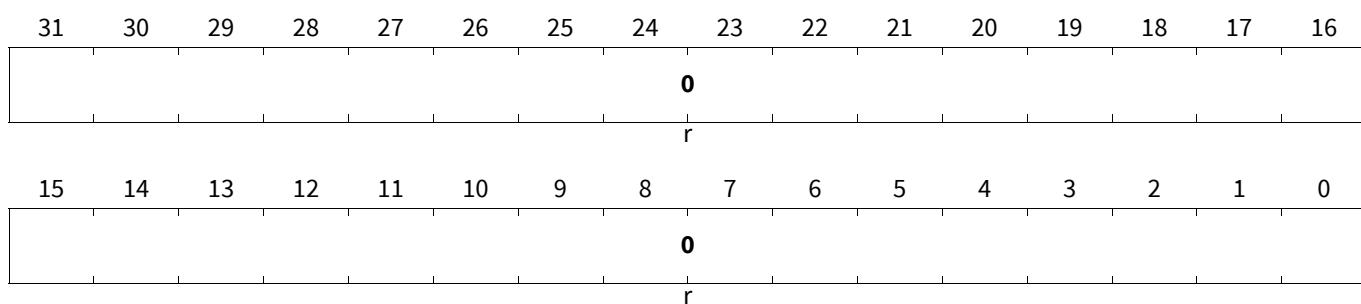
## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the PMS kernel addresses for transactions with the Master TAG ID n 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

### Access Enable Register 1

#### ACCEN1

**Access Enable Register 1** **Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>0</b>	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

### 12.3.2.6 SMU Registers

The following registers are specified in the SMU chapter of this book:

- AG2i\_STDBY (i=0)
- MONBISTSTAT
- MONBISTCTRL
- CMD\_STDBY
- AG2iFSP\_STDBY (i=0)

## Power Management System for Low-End (PMSLE)

### 12.3.3 Power Management Control Registers (SCU)

**Table 471 Register Overview - PMC (sorted by Name)**

Short Name	Long Name	Offset Address	Page Number
DTSCLIM	Core Die Temperature Sensor Limit Register	0108 <sub>H</sub>	<a href="#">185</a>
DTSCSTAT	Core Die Temperature Sensor Status Register	0104 <sub>H</sub>	<a href="#">185</a>
PMCSR0	Power Management Control and Status Register	00C8 <sub>H</sub>	<a href="#">177</a>
PMCSR1	Power Management Control and Status Register	00CC <sub>H</sub>	<a href="#">178</a>
PMCSR2	Power Management Control and Status Register	00D0 <sub>H</sub>	<a href="#">179</a>
PMCSR3	Power Management Control and Status Register	00D4 <sub>H</sub>	<a href="#">180</a>
PMCSR4	Power Management Control and Status Register	00D8 <sub>H</sub>	<a href="#">181</a>
PMCSR5	Power Management Control and Status Register	00DC <sub>H</sub>	<a href="#">182</a>
PMSTAT0	Power Management Status Register 0	00E4 <sub>H</sub>	<a href="#">175</a>
PMSWCR1	Standby and Wake-up Control Register 1	00E8 <sub>H</sub>	<a href="#">183</a>
PMTRCSR0	Power Management Transition Control and Status Register 0	0198 <sub>H</sub>	<a href="#">187</a>
PMTRCSR1	Power Management Transition Control and Status Register 1	019C <sub>H</sub>	<a href="#">189</a>
PMTRCSR2	Power Management Transition Control and Status Register 2	01A0 <sub>H</sub>	<a href="#">190</a>
PMTRCSR3	Power Management Transition Control and Status Register 3	01A4 <sub>H</sub>	<a href="#">191</a>

#### 12.3.3.1 Power Management Control and Status Registers

This section describes the kernel registers of the PMS module in SCU address space. Most of PMS kernel register names described in this section will be referenced in other parts of the Target Specification by the module name prefix “SCU\_”. The set of registers used for Power Management control the issue of power modes, manage wake-up configuration and provide status information on mode transitions and modules. The request for Idle, Sleep or Standby mode is issued via PMCSR<sub>x</sub> registers.

Tricore atomic instructions (LDMST, ST.T, SWAP.W, SWAPMASK.W, CMPSWAP.W) only write back bits that are changing their level. This leads to the fact that bits that are already set cannot be written with a 1 when using RMW instructions. No problem exists when using direct write instructions (e.g. ST.W). This affects the status bits in register DTSCSTAT.LLU, UOF and INT bits which are cleared by writing 1s.

##### Power Management Status Register 0

###### PMSTAT0

Power Management Status Register 0 ( <a href="#">00E4<sub>H</sub></a> ) Application Reset Value: 0000 0001 <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

0

r															

CPU3L  
S

rh															

CPU2L  
S

rh															

CPU1L  
S

rh															

CPU0L  
S

rh															

0

r															

CPU5  
CPU4  
CPU3  
CPU2  
CPU1  
CPU0

rh															

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>CPU0</b>	0	rh	<b>CPU0 Status</b> This bit field reflects the current status of CPU0. 0 <sub>B</sub> CPU0 is in Halt or Idle Mode 1 <sub>B</sub> CPU0 is in Normal Run Mode
<b>CPU1</b>	1	rh	<b>CPU1 Status</b> This bit field reflects the current status of CPU1. 0 <sub>B</sub> CPU1 is in Halt or Idle Mode 1 <sub>B</sub> CPU1 is in Normal Run Mode
<b>CPU2</b>	2	rh	<b>CPU2 Status</b> This bit field reflects the current status of CPU2. 0 <sub>B</sub> CPU2 is in Halt or Idle Mode 1 <sub>B</sub> CPU2 is in Normal Run Mode
<b>CPU3</b>	3	rh	<b>CPU3 Status</b> This bit field reflects the current status of CPU3. 0 <sub>B</sub> CPU3 is in Halt or Idle Mode 1 <sub>B</sub> CPU3 is in Normal Run Mode
<b>CPU4</b>	4	rh	<b>CPU4 Status</b> This bit field reflects the current status of CPU4. 0 <sub>B</sub> CPU4 is in Halt or Idle Mode 1 <sub>B</sub> CPU4 is in Normal Run Mode
<b>CPU5</b>	5	rh	<b>CPU5 Status</b> This bit field reflects the current status of CPU5. 0 <sub>B</sub> CPU5 is in Halt or Idle Mode 1 <sub>B</sub> CPU5 is in Normal Run Mode
<b>CPU0LS</b>	16	rh	<b>CPU0LS Status</b> This bit field reflects the current status of CPU0 Lockstep Checker Core. The activation of the Lockstep is configured in UCB BMI configuration and determines the default reset value. The default reset value 0 is for the case where CPU0LS is disabled in UCB BMI configuration. 0 <sub>B</sub> CPU0LS is disabled or in Halt or Idle Mode 1 <sub>B</sub> CPU0LS is enabled and in Normal Run Mode
<b>CPU1LS</b>	17	rh	<b>CPU1LS Status</b> This bit field reflects the current status of CPU1 Lockstep Checker Core. The activation of the Lockstep is configured in UCB BMI configuration and determines the default status. The default reset value 0 is for the case where CPU1LS is disabled in UCB BMI configuration. 0 <sub>B</sub> CPU1LS is disabled or in Halt or Idle Mode 1 <sub>B</sub> CPU1LS is enabled and in Normal Run Mode

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
CPU2LS	18	rh	<b>CPU2LS Status</b> This bit field reflects the current status of CPU2 Lockstep Checker Core. The activation of the Lockstep is configured in UCB BMI configuration and determines the default status. The default reset value 0 is for the case where CPU0LS is disabled in UCB BMI configuration. 0 <sub>B</sub> CPU2LS is disabled or in Halt or Idle Mode 1 <sub>B</sub> CPU2LS is enabled and in Normal Run Mode
CPU3LS	19	rh	<b>CPU3LS Status</b> This bit field reflects the current status of CPU3 Lockstep Checker Core. The activation of the Lockstep is configured in UCB BMI configuration and determines the default status. The default reset value 0 is for the case where CPU0LS is disabled in UCB BMI configuration. 0 <sub>B</sub> CPU3LS is disabled or in Halt or Idle Mode 1 <sub>B</sub> CPU3LS is enabled and in Normal Run Mode
0	15:6, 31:20	r	<b>Reserved</b> Read as 0; should be written with 0.

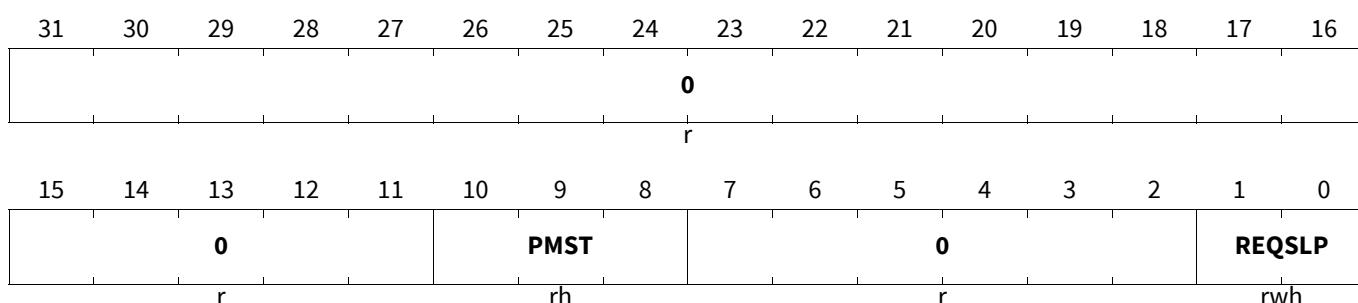
## Power Management Control and Status Register

Power Management Control and Status Register for CPU0

### PMCSR0

#### Power Management Control and Status Register(00C8<sub>H</sub>)

Application Reset Value: 0000 0100<sub>H</sub>



Field	Bits	Type	Description
REQSLP	1:0	rwh	<b>Idle Mode and Sleep Mode Request</b> In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUXSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. REQSLP maybe written only when either CPU or Safety ENDINIT bits are set to 0. CPU ENDINIT bit has to be set back after REQSLP is written for the mode transition to take place. In case of Safety ENDINIT, the mode transition will be issued immediately and does not wait till Safety ENDINIT is set back to 1 again. 00 <sub>B</sub> Request CPU Run Mode 01 <sub>B</sub> Request CPU Idle Mode 10 <sub>B</sub> Request System Sleep Mode 11 <sub>B</sub> Request System Standby Mode

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>PMST</b>	10:8	rh	<b>Power management Status</b> This bit field reflects the current status of the CPU. 000 <sub>B</sub> Reserved, do not use this combination 001 <sub>B</sub> Normal Run Mode <sup>1)</sup> 010 <sub>B</sub> CPU Idle Mode requested 011 <sub>B</sub> CPU Idle Mode acknowledged 100 <sub>B</sub> Sleep Mode requested 101 <sub>B</sub> Reserved, do not use this combination 110 <sub>B</sub> Standby Mode requested 111 <sub>B</sub> Reserved, do not use this combination
<b>0</b>	7:2, 31:11	r	<b>Reserved</b> Read as 0; should be written with 0.

- 1) After a reset, all CPUs are in “Normal Run Mode”, but this does not mean that all CPUs are executing code. This mode also includes the CPU “halt” mode which is the start-up default for all except CPU0.

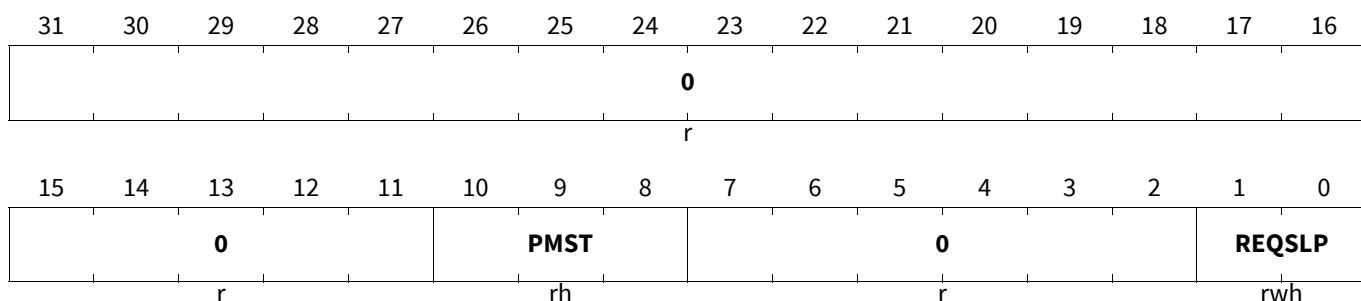
## Power Management Control and Status Register

Power Management Control and Status Register for CPU1. On product variants where CPU1 is not available, this register has no function.

### PMCSR1

#### Power Management Control and Status Register(00CC<sub>H</sub>)

Application Reset Value: 0000 0100<sub>H</sub>



Field	Bits	Type	Description
<b>REQSLP</b>	1:0	rwh	<b>Idle Mode and Sleep Mode Request</b> In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. REQSLP maybe written only when either CPU or Safety ENDINIT bits are set to 0. CPU ENDINIT bit has to be set back after REQSLP is written for the mode transition to take place. In case of Safety ENDINIT, the mode transition will be issued immediately and does not wait till Safety ENDINIT is set back to 1 again. 00 <sub>B</sub> Request CPU Run Mode 01 <sub>B</sub> Request CPU Idle Mode 10 <sub>B</sub> Request System Sleep Mode 11 <sub>B</sub> Request System Standby Mode

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>PMST</b>	10:8	rh	<b>Power management Status</b> This bit field reflects the current status of the CPU. 000 <sub>B</sub> Reserved, do not use this combination 001 <sub>B</sub> Normal Run Mode <sup>1)</sup> 010 <sub>B</sub> CPU Idle Mode requested 011 <sub>B</sub> CPU Idle Mode acknowledged 100 <sub>B</sub> Sleep Mode requested 101 <sub>B</sub> Reserved, do not use this combination 110 <sub>B</sub> Standby Mode requested 111 <sub>B</sub> Reserved, do not use this combination
<b>0</b>	7:2, 31:11	r	<b>Reserved</b> Read as 0; should be written with 0.

- 1) After a reset, all CPUs are in “Normal Run Mode”, but this does not mean that all CPUs are executing code. This mode also includes the CPU “halt” mode which is the start-up default for all except CPU0.

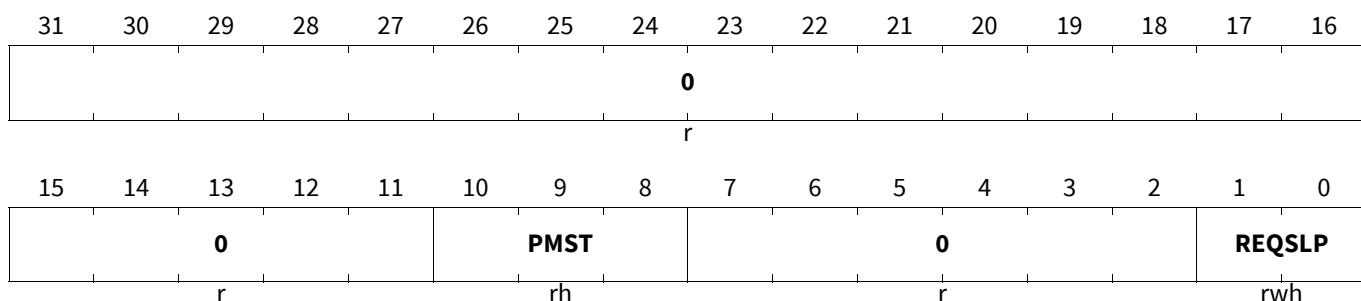
## Power Management Control and Status Register

Power Management Control and Status Register for CPU2. On product variants where CPU2 is not available, this register has no function.

### PMCSR2

#### Power Management Control and Status Register(00D0<sub>H</sub>)

Application Reset Value: 0000 0100<sub>H</sub>



Field	Bits	Type	Description
<b>REQSLP</b>	1:0	rwh	<b>Idle Mode and Sleep Mode Request</b> In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. REQSLP maybe written only when either CPU or Safety ENDINIT bits are set to 0. CPU ENDINIT bit has to be set back after REQSLP is written for the mode transition to take place. In case of Safety ENDINIT, the mode transition will be issued immediately and does not wait till Safety ENDINIT is set back to 1 again. 00 <sub>B</sub> Request CPU Run Mode 01 <sub>B</sub> Request CPU Idle Mode 10 <sub>B</sub> Request System Sleep Mode 11 <sub>B</sub> Request System Standby Mode

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>PMST</b>	10:8	rh	<b>Power management Status</b> This bit field reflects the current status of the CPU. 000 <sub>B</sub> Reserved, do not use this combination 001 <sub>B</sub> Normal Run Mode <sup>1)</sup> 010 <sub>B</sub> CPU Idle Mode requested 011 <sub>B</sub> CPU Idle Mode acknowledged 100 <sub>B</sub> Sleep Mode requested 101 <sub>B</sub> Reserved, do not use this combination 110 <sub>B</sub> Standby Mode requested 111 <sub>B</sub> Reserved, do not use this combination
<b>0</b>	7:2, 31:11	r	<b>Reserved</b> Read as 0; should be written with 0.

- 1) After a reset, all CPUs are in “Normal Run Mode”, but this does not mean that all CPUs are executing code. This mode also includes the CPU “halt” mode which is the start-up default for all except CPU0.

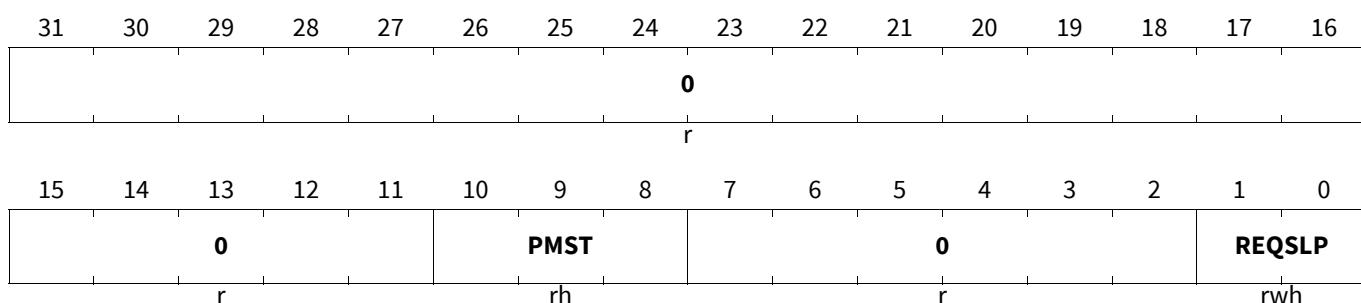
## Power Management Control and Status Register

Power Management Control and Status Register for CPU3. On product variants where CPU3 is not available, this register has no function.

### PMCSR3

#### Power Management Control and Status Register(00D4<sub>H</sub>)

Application Reset Value: 0000 0100<sub>H</sub>



Field	Bits	Type	Description
<b>REQSLP</b>	1:0	rwh	<b>Idle Mode and Sleep Mode Request</b> In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. REQSLP maybe written only when either CPU or Safety ENDINIT bits are set to 0. CPU ENDINIT bit has to be set back after REQSLP is written for the mode transition to take place. In case of Safety ENDINIT, the mode transition will be issued immediately and does not wait till Safety ENDINIT is set back to 1 again. 00 <sub>B</sub> Request CPU Run Mode 01 <sub>B</sub> Request CPU Idle Mode 10 <sub>B</sub> Request System Sleep Mode 11 <sub>B</sub> Request System Standby Mode

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>PMST</b>	10:8	rh	<b>Power management Status</b> This bit field reflects the current status of the CPU. 000 <sub>B</sub> Reserved, do not use this combination 001 <sub>B</sub> Normal Run Mode <sup>1)</sup> 010 <sub>B</sub> CPU Idle Mode requested 011 <sub>B</sub> CPU Idle Mode acknowledged 100 <sub>B</sub> Sleep Mode requested 101 <sub>B</sub> Reserved, do not use this combination 110 <sub>B</sub> Standby Mode requested 111 <sub>B</sub> Reserved, do not use this combination
<b>0</b>	7:2, 31:11	r	<b>Reserved</b> Read as 0; should be written with 0.

- 1) After a reset, all CPUs are in “Normal Run Mode”, but this does not mean that all CPUs are executing code. This mode also includes the CPU “halt” mode which is the start-up default for all except CPU0.

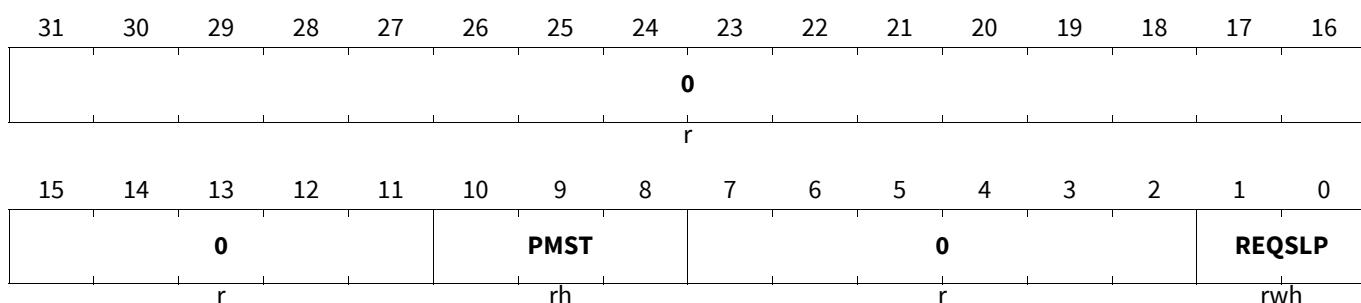
## Power Management Control and Status Register

Power Management Control and Status Register for CPU4. On product variants where CPU4 is not available, this register has no function.

### PMCSR4

#### Power Management Control and Status Register(00D8<sub>H</sub>)

Application Reset Value: 0000 0100<sub>H</sub>



Field	Bits	Type	Description
<b>REQSLP</b>	1:0	rwh	<b>Idle Mode and Sleep Mode Request</b> In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. REQSLP maybe written only when either CPU or Safety ENDINIT bits are set to 0. CPU ENDINIT bit has to be set back after REQSLP is written for the mode transition to take place. In case of Safety ENDINIT, the mode transition will be issued immediately and does not wait till Safety ENDINIT is set back to 1 again. 00 <sub>B</sub> Request CPU Run Mode 01 <sub>B</sub> Request CPU Idle Mode 10 <sub>B</sub> Request System Sleep Mode 11 <sub>B</sub> Request System Standby Mode

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>PMST</b>	10:8	rh	<b>Power management Status</b> This bit field reflects the current status of the CPU. 000 <sub>B</sub> Reserved, do not use this combination 001 <sub>B</sub> Normal Run Mode <sup>1)</sup> 010 <sub>B</sub> CPU Idle Mode requested 011 <sub>B</sub> CPU Idle Mode acknowledged 100 <sub>B</sub> Sleep Mode requested 101 <sub>B</sub> Reserved, do not use this combination 110 <sub>B</sub> Standby Mode requested 111 <sub>B</sub> Reserved, do not use this combination
<b>0</b>	7:2, 31:11	r	<b>Reserved</b> Read as 0; should be written with 0.

- 1) After a reset, all CPUs are in “Normal Run Mode”, but this does not mean that all CPUs are executing code. This mode also includes the CPU “halt” mode which is the start-up default for all except CPU0.

## Power Management Control and Status Register

Power Management Control and Status Register for CPU5. On product variants where CPU5 is not available, this register has no function.

### PMCSR5

#### Power Management Control and Status Register(00DC<sub>H</sub>)

Application Reset Value: 0000 0100<sub>H</sub>



Field	Bits	Type	Description
<b>REQSLP</b>	1:0	rwh	<b>Idle Mode and Sleep Mode Request</b> In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. REQSLP maybe written only when either CPU or Safety ENDINIT bits are set to 0. CPU ENDINIT bit has to be set back after REQSLP is written for the mode transition to take place. In case of Safety ENDINIT, the mode transition will be issued immediately and does not wait till Safety ENDINIT is set back to 1 again. 00 <sub>B</sub> Request CPU Run Mode 01 <sub>B</sub> Request CPU Idle Mode 10 <sub>B</sub> Request System Sleep Mode 11 <sub>B</sub> Request System Standby Mode

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
PMST	10:8	rh	<b>Power management Status</b> This bit field reflects the current status of the CPU. 000 <sub>B</sub> Reserved, do not use this combination 001 <sub>B</sub> Normal Run Mode <sup>1)</sup> 010 <sub>B</sub> CPU Idle Mode requested 011 <sub>B</sub> CPU Idle Mode acknowledged 100 <sub>B</sub> Sleep Mode requested 101 <sub>B</sub> Reserved, do not use this combination 110 <sub>B</sub> Standby Mode requested 111 <sub>B</sub> Reserved, do not use this combination
0	7:2, 31:11	r	<b>Reserved</b> Read as 0; should be written with 0.

- 1) After a reset, all CPUs are in “Normal Run Mode”, but this does not mean that all CPUs are executing code. This mode also includes the CPU “halt” mode which is the start-up default for all except CPU0.

## Standby and Wake-up Control Register 1

### PMSWCR1

#### Standby and Wake-up Control Register 1

(00E8<sub>H</sub>)

Cold PowerOn Reset Value: 0100 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	<b>STBYEV</b>			<b>STBYEVEN</b>	<b>CPUSEL</b>			0							
r	rw			w	rw							r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	<b>IRADIS</b>		<b>0</b>	<b>CPUIDLSEL</b>			0								
r	rw		r	rw								r			

Field	Bits	Type	Description
CPUIDLSEL	10:8	rw	<b>CPU selection for Idle mode</b> This bit field allows a CPUx to issue Idle request to other CPUs in addition to itself. A request for Idle via PMCSR <sub>x</sub> .REQSLP=01 by CPUx will also trigger Idle requests to all other CPUs. <i>Note:</i> All other CPUIDLSEL bit combinations are reserved. 000 <sub>B</sub> Entry to the respective Idle mode is decided by each individual CPU. 001 <sub>B</sub> CPU0 Idle request will send all CPUs in Idle. 010 <sub>B</sub> CPU1 Idle request will send all CPUs in Idle. 011 <sub>B</sub> CPU2 Idle request will send all CPUs in Idle. 100 <sub>B</sub> CPU3 Idle request will send all CPUs in Idle. 101 <sub>B</sub> CPU4 Idle request will send all CPUs in Idle. 110 <sub>B</sub> CPU5 Idle request will send all CPUs in Idle.

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>IRADIS</b>	12	rw	<p><b>Idle-Request-Acknowledge Sequence Disable</b></p> <p>This bit enables SCU Idle Request Acknowledge sequence to all modules on Standby entry. IRADIS bit has no effect incase of Standby entry triggered via PWRWKEN register bit.</p> <p>This bit shall be set before Standby entry to disable Idle request acknowledge sequence so that standby request is not blocked by a pending reset idle request acknowledge sequence.</p> <p><math>0_B</math> Idle-Request-Acknowledge Sequence issued on Standby entry.  <math>1_B</math> Idle-Request-Acknowledge Sequence skipped on Standby entry.</p>
<b>CPUSEL</b>	26:24	rw	<p><b>CPU selection for Sleep and Standby mode</b></p> <p><i>Note:</i> All other CPUSEL bit combinations are reserved.</p> <p><math>001_B</math> Only CPU0 can trigger power down modes.  <math>010_B</math> Only CPU1 can trigger power down modes.  <math>011_B</math> Only CPU2 can trigger power down modes.  <math>100_B</math> Only CPU3 can trigger power down modes.  <math>101_B</math> Only CPU4 can trigger power down modes.  <math>110_B</math> Only CPU5 can trigger power down modes.  <math>111_B</math> Entry to power down modes is unanimously decided by all the CPUs.</p>
<b>STBYEVEN</b>	27	w	<p><b>Standby Entry Event configuration enable</b></p> <p><math>0_B</math> Bit STBYEV is not updated.  <math>1_B</math> Bit STBYEV can be updated.</p>
<b>STBYEV</b>	30:28	rw	<p><b>Standby Entry Event Configuration</b></p> <p><i>Note:</i> All other bit combinations are reserved.</p> <p><math>000_B</math> Standby Entry triggered by setting PMCSR<sub>x</sub>.REQSLP register bit (Default).  <math>100_B</math> Standby Entry triggered on ESR1 / NMI assertion.</p>
<b>0</b>	7:0, 11, 23:13, 31	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

## Power Management System for Low-End (PMSLE)

### Core Die Temperature Sensor Status Register

#### DTSCSTAT

Core Die Temperature Sensor Status Register (0104<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RESULT							
r								rh							

Field	Bits	Type	Description
RESULT	11:0	rh	<b>Result of the DTSC Measurement</b> This bit field shows the result of the DTSC measurement. The value given is directly related to the die temperature and can be evaluated using the following formula. $T (\text{°C}) = [\text{RESULT} / \text{Gnom}] - 273.15$ $T (\text{°K}) = [\text{RESULT}] / \text{G\_nom}$ $\text{RESULT} = \text{G\_nom} * \{T (\text{°C}) + 273.15\} = \text{G\_nom} * T (\text{°K})$ $\text{G\_nom} = 7.505$
0	31:12	r	<b>Reserved</b> Read as 0.

### Core Die Temperature Sensor Limit Register

#### DTSCLIM

Core Die Temperature Sensor Limit Register (0108<sub>H</sub>)

Application Reset Value: 0CD8 06D6<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UOF INT 0 INTEN															
rwh rwh r rw rw rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLU EN BGPO K 0								LOWER							
rwh rw rh r rw								rw							

Field	Bits	Type	Description
LOWER	11:0	rw	<b>DTSC Lower Limit</b> This bit field defines the lower limit of the DTSC temperature check. The DTSC measurement result is compared against this value and if the measurement result is less than or equal to the configured LOWER bitfield value; flag LLU is set.

**Power Management System for Low-End (PMSLE)**

Field	Bits	Type	Description
<b>BGPOK</b>	13	rh	<b>DTSC Bandgap OK</b> This bitfield indicates that the bandgap reference for the Core Die Temperature Sensor (DTSC) is available and ok. 0 <sub>B</sub> DTSC Bandgap is not ok. 1 <sub>B</sub> DTSC Bandgap is ok.
<b>EN</b>	14	rw	<b>DTSC Enable</b> This bitfield enables the Core Die Temperature Sensor (DTSC). The bitfield is reset on an application reset. 0 <sub>B</sub> DTSC is disabled 1 <sub>B</sub> DTSC is enabled
<b>LLU</b>	15	rwh	<b>DTSC Lower Limit Underflow</b> When this bit is set the related SMU DTSC alarm trigger is generated. This bit has to be written with zero in order to clear it. Writing a one has no effect. This bit is set when a DTSC measurement is finished and the result is below the lower limit (i.e. DTSCLIM.LOWER). 0 <sub>B</sub> No temperature underflow was detected 1 <sub>B</sub> A temperature underflow was detected
<b>UPPER</b>	27:16	rw	<b>DTSC Upper Limit</b> This bit field defines the upper limit of the DTSC temperature check. The DTSC measurement result is compared against this value and if the measurement result is greater than or equal to the configured UPPER bitfield value; flag UOF is set.
<b>INTEN</b>	28	rw	<b>DTSC Interrupt Enable</b> This bitfield enables the Core Die Temperature Sensor (DTSC) interrupt. The bitfield is reset on an application reset. 0 <sub>B</sub> DTSC Interrupt is disabled 1 <sub>B</sub> DTSC Interrupt is enabled
<b>INT</b>	30	rwh	<b>DTSC Interrupt status flag</b> This bit is set when SMU DTSC interrupt is generated when a DTSC measurement is finished. This bit is cleared by writing a zero. Writing a one has no effect. 0 <sub>B</sub> No DTSC interrupt is generated 1 <sub>B</sub> DTSC interrupt is generated
<b>UOF</b>	31	rwh	<b>DTSC Upper Limit Overflow</b> When this bit is set, the related SMU DTSC alarm trigger is generated. This bit has to be written with zero in order to clear it. Writing a one has no effect. This bit is set when a DTSC measurement is finished and the result is exceeding the upper limit (i.e. DTSCLIM.UPPER). 0 <sub>B</sub> No temperature overflow was detected 1 <sub>B</sub> A temperature overflow was detected
<b>0</b>	12, 29	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System for Low-End (PMSLE)

### Power Management Transition Control and Status Register 0

#### PMTRCSR0

**Power Management Transition Control and Status Register 0(0198<sub>H</sub>)**

**Cold PowerOn Reset Value: 0000**

**0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
										<b>VDTCL R</b>	<b>VDTST P</b>	<b>VDTST RT</b>	<b>VDTO VIEN</b>	<b>VDTO VEN</b>	<b>VDTEN</b>
0	LPSLP EN				0					w	rw	rwh	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										<b>LJTCL R</b>	<b>LJTST P</b>	<b>LJTST RT</b>	<b>LJTOV IEN</b>	<b>LJTOV EN</b>	<b>LJTEN</b>
	SDSTEP					0				w	rw	rwh	rw	rw	rw

Field	Bits	Type	Description
<b>LJTEN</b>	0	rw	<b>Load Jump Timer Enable</b> This bit field enables the usage of load jump timer. 0 <sub>B</sub> Load Jump Timer inactive 1 <sub>B</sub> Load Jump Timer active
<b>LJTOVEN</b>	1	rw	<b>Load Jump Timer Overflow Enable</b> This bit field enables the update of LJTOV status bit on timer overflow or time out. 0 <sub>B</sub> LJTOV bit is not updated on a Load Jump Timer overflow. 1 <sub>B</sub> LJTOV bit is updated on a Load Jump Timer overflow.
<b>LJTOVIEN</b>	2	rw	<b>Load Jump Timer Overflow Interrupt Enable</b> This bit field enables the activation of interrupt on timer overflow or time out. 0 <sub>B</sub> LJTOV interrupt is inactive. 1 <sub>B</sub> LJTOV interrupt is activated on a Load Jump Timer overflow.
<b>LJTSTRT</b>	3	rwh	<b>Load Jump Timer Start</b> This bit field starts Load jump timer. This is intended for test purposes. The LJTSTRT remains set on a write and is cleared when LJTOV bit is set if LJTOVEN bit is enabled. 0 <sub>B</sub> Load Jump Timer status not changed. 1 <sub>B</sub> Load Jump Timer started.
<b>LJTSTP</b>	4	rw	<b>Load Jump Timer Stop</b> This bit field stops Load jump timer. This is intended for test purposes. The LJTSTP remains set on a write and is to be explicitly cleared by software. The LJTSTP stops the counter at the current value and timer re-starts from that value when LJTSTP is cleared and LJTSTRT is set. 0 <sub>B</sub> Load Jump Timer status not changed. 1 <sub>B</sub> Load Jump Timer stopped.

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
LJTCLR	5	w	<b>Load Jump Timer Clear</b> This bit field clear Load jump timer count. This is intended for test purposes. This bit resets LJT and clears LJTRUN if LJTEN bit is set. 0 <sub>B</sub> Load Jump Count status not changed. 1 <sub>B</sub> Load Jump Timer Count cleared.
SDSTEP	15:12	rw	<b>Droop Voltage Step(vdroop_step_i)</b> This bit field defines the voltage offset for droop compensation on a load jump to the EVRC setpoint value. The request is made via <b>PMTRCSR3.VDROOPREQ</b> on an anticipated load jump with a voltage offset equal to the SDSTEP x 5 mV. The droop step is a positive offset if VDROOPREQ = 01b and is a negative offset if VDROOPREQ = 10b and no offset is applied if VDROOPREQ = 00b. Maximum Droop = 80 mV.
VDTEN	16	rw	<b>Voltage Droop Timer Enable</b> This bit field enables the usage of Voltage Droop timer. 0 <sub>B</sub> Voltage Droop Timer inactive 1 <sub>B</sub> Voltage Droop Timer active
VDTOVEN	17	rw	<b>Voltage Droop Timer Overflow Enable</b> This bit field enables the update of VDTOV status bit on timer overflow or time out. 0 <sub>B</sub> VDTOV bit is not updated on a Voltage Droop Timer overflow. 1 <sub>B</sub> VDTOV bit is updated on a Voltage Droop Timer overflow.
VDTOVIEN	18	rw	<b>Voltage Droop Timer Overflow Interrupt Enable</b> This bit field enables the activation of interrupt on timer overflow or time out. 0 <sub>B</sub> VDTOV interrupt is inactive. 1 <sub>B</sub> VDTOV interrupt is activated on a Voltage Droop Timer overflow.
VDTSTRT	19	rwh	<b>Voltage Droop Timer Start</b> This bit field starts Voltage Droop timer. This is intended for test purposes. The VDTSTRT remains set on a write and is cleared when VDTOV bit is set if VDTOVEN bit is enabled. 0 <sub>B</sub> Voltage Droop Timer status not changed. 1 <sub>B</sub> Voltage Droop Timer started.
VDTSTP	20	rw	<b>Voltage Droop Timer Stop</b> This bit field stops Voltage Droop timer. SCU cancels the droop request via signal sd_droop_cntr_i = 00. This is intended for test purposes. The VDTSTP remains set on a write and is to be explicitly cleared by software. The VDTSTP stops the counter at the current value and timer re-starts from that value when VDTSTP is cleared and VDTSTRT is set. 0 <sub>B</sub> Voltage Droop Timer status not changed. 1 <sub>B</sub> Voltage Droop Timer stopped.
VDTCLR	21	w	<b>Voltage Droop Timer Clear</b> This bit field clear Voltage Droop timer count. This is intended for test purposes. This bit resets VDT and clears VDTRUN if VDTEN bit is set. 0 <sub>B</sub> Voltage Droop Count status not changed. 1 <sub>B</sub> Voltage Droop Timer Count cleared.

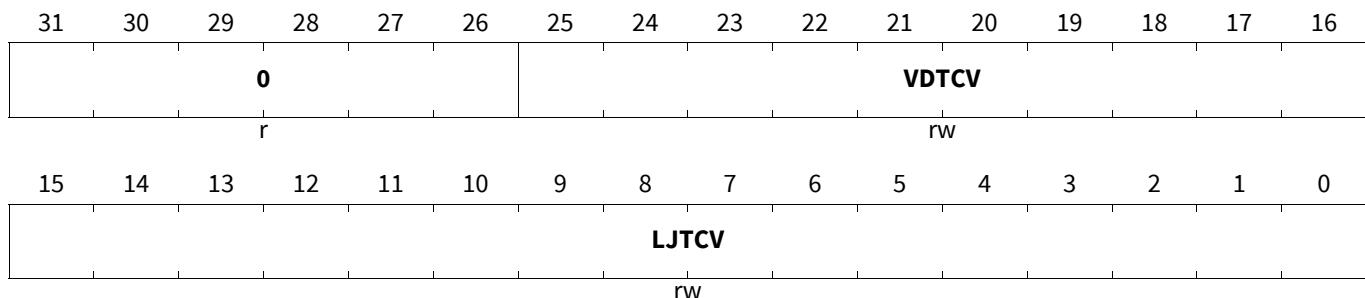
## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
LPSLPEN	29	rw	<b>EVRC Low Power Mode activation on a Sleep Request</b> PMS: This bit field enables the activation of LPM EVRC mode on a sleep request. PMSLE: Reserved, no function (no LPM for SC-DCDC EVRC). 0 <sub>B</sub> PMS: EVRC remains in normal operation mode during and after a sleep request. PMSLE: Reserved. 1 <sub>B</sub> PMS: LPM mode activated on a sleep request. PMSLE: Reserved.
0	11:6, 28:22, 31:30	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management Transition Control and Status Register 1

### PMTRCSR1

**Power Management Transition Control and Status Register 1(019C<sub>H</sub>)**      **Cold PowerOn Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
LJTCV	15:0	rw	<b>Load Jump Timer Compare Setpoint Value</b> This bit field defines the compare setpoint value of Load Jump timer. The compare event would lead to LJTOV bit being set and LJT interrupt being raised. The LJTRUN status bit, LDJMPREQ bit and LJTCNT value is reset to 0 on a compare event. X us is the compare value. LSB =1 us. Total range = 65.5 ms
VDTCV	25:16	rw	<b>Voltage Droop Timer Compare Setpoint Value</b> This bit field defines the compare setpoint value of Voltage Droop timer. The compare event would lead to VDTOV bit being set and VDT interrupt being raised. The VDTRUN status bit, VDROOPREQ bit and VDTCNT value is reset to 0 on a compare event. X us is the compare value. LSB =1 us. Total range = 1023 us
0	31:26	r	<b>Reserved</b> Read as 0; should be written with 0.

## Power Management System for Low-End (PMSLE)

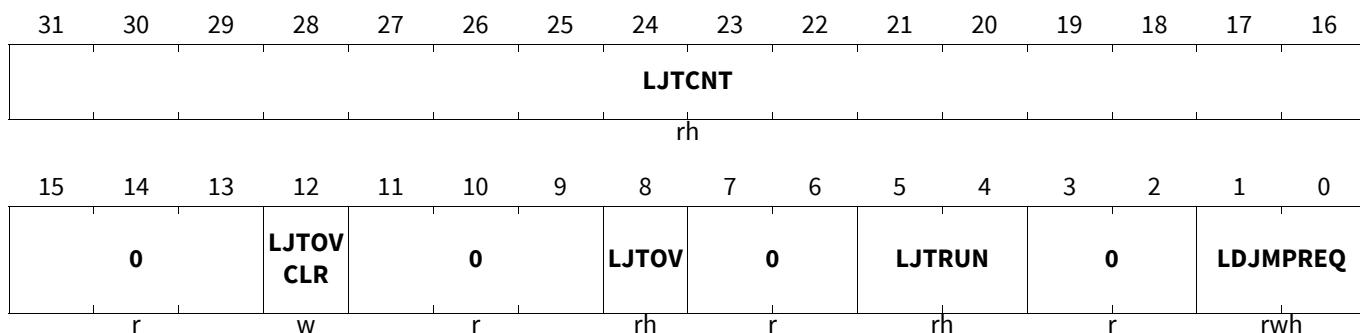
### Power Management Transition Control and Status Register 2

#### PMTRCSR2

**Power Management Transition Control and Status Register 2(01A0<sub>H</sub>)**

**Cold PowerOn Reset Value: 0000**

**0000<sub>H</sub>**



Field	Bits	Type	Description
<b>LDJMPREQ</b>	1:0	rwh	<p><b>Load Jump Request</b></p> <p>This bit requests a Load Jump consequently leading to Load Jump Timer start and LJTRUN bit being set if LJTEN=1. The request is not taken if LJTRUN bit is already in set state and LJT is currently running. The request is not taken if VDTRUN bit is already in set state and VDT is currently running. The request is also not taken if (LJTOV bit is set AND LJTOVEN bit is enabled). The request is also not taken if (VDTOV bit is set AND VDTOVEN bit is enabled). The LDJMPREQ bit is cleared on a compare overflow.</p> <p><i>Note:</i> All other bit combinations are reserved.</p> <p>00<sub>B</sub> Load Jump Timer inactive 01<sub>B</sub> Load Jump Request made and taken. Load Jump Timer activated.</p>
<b>LJTRUN</b>	5:4	rh	<p><b>Load Jump Timer Run Status</b></p> <p>This status bit indicates that the Load Jump timer is currently running and a Load Jump is currently taking place. The LJTRUN bit is cleared on a compare overflow.</p> <p><i>Note:</i> All other bit combinations are reserved.</p> <p>00<sub>B</sub> Load Jump and Load Jump Timer inactive 01<sub>B</sub> A SW triggered Load Jump active and Load Jump Timer active 10<sub>B</sub> A HW triggered Load Jump active and Load Jump Timer active (reserved for future)</p>

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
LJTOV	8	rh	<b>Load Jump Timer Overflow Status</b> This status bit indicates that the Load Jump timer compare match has happened. If LJTOVEN bit is enabled, then LJTOV can only be cleared explicitly via LJTOVCLR bit. If LJTOVEN bit is disabled, LJTOV is cleared on a taken Load Jump Request (A new Load Jump request is taken only if both LJT & VDT are not currently running and no active Load Jump request is being processed). LJTOV being set will lead to an interrupt if LJTOVIEN is enabled. 0 <sub>B</sub> Load Jump Timer compare overflow has not happened. 1 <sub>B</sub> Load Jump Timer compare overflow has happened.
LJTOVCLR	12	w	<b>Load Jump Timer Overflow Status Clear</b> This bit clears LJTOV status bit and sets VDROOPREQ and LDJMPREQ to 0 if LJTOVEN bit is enabled. This bit always reads as 0. 0 <sub>B</sub> This clear bit has no effect on Load Jump Timer overflow flag. 1 <sub>B</sub> Load Jump Timer overflow flag is cleared.
LJTCNT	31:16	rh	<b>Load Jump Timer Value</b> This bit field reflects the current Load Jump timer value. LJTCNT value is cleared on timer overflow and on a taken Load Jump Request X us is the compare value. LSB = 1 us. Total range = 65.5 ms
0	3:2, 7:6, 11:9, 15:13	r	<b>Reserved</b> Read as 0; should be written with 0.

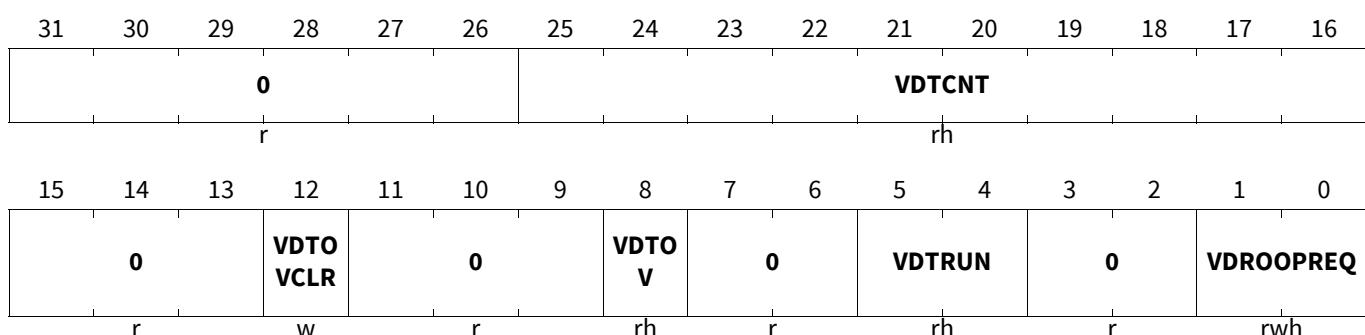
## Power Management Transition Control and Status Register 3

### PMTRCSR3

#### Power Management Transition Control and Status Register 3(01A4<sub>H</sub>)

Cold PowerOn Reset Value: 0000

0000<sub>H</sub>



## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
<b>VDROOPREQ</b>	1:0	rwh	<p><b>Voltage Droop Request</b></p> <p>This bit requests a Voltage Droop consequently leading to Voltage Droop Timer start and VDTRUN bit being set if VDTEN=1. The request is not taken if VDTRUN bit is already in set state and VDT is currently running. The request is also not taken if (VDTOV bit is set AND VDTOVEN bit is enabled). The droop step is a positive offset if sd_droop_cntr_i = 01 and is a negative offset if sd_droop_cntr_i = 10 and no offset is applied if sd_droop_cntr_i = 00 and is applied immediately.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Voltage Droop and Voltage Droop Timer inactive</li> <li>01<sub>B</sub> A Positive Voltage Droop Request made and taken. Voltage Droop Timer activated.</li> <li>10<sub>B</sub> A Negative Voltage Droop Request made and taken. Voltage Droop Timer activated.</li> <li>11<sub>B</sub> Voltage Droop and Voltage Droop Timer inactive</li> </ul>
<b>VDTRUN</b>	5:4	rh	<p><b>Voltage Droop Timer Run Status</b></p> <p>This status bit indicates that the Voltage Droop timer is currently running and a Voltage Droop is currently taking place. The VDTRUN bit is cleared on a compare overflow.</p> <p><i>Note:</i> All other bit combinations are reserved.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> Voltage Droop and Voltage Droop Timer inactive</li> <li>01<sub>B</sub> A SW triggered Voltage Droop active and Voltage Droop Timer active</li> <li>10<sub>B</sub> A HW triggered Voltage Droop active and Voltage Droop Timer active (reserved for future)</li> </ul>
<b>VDTOV</b>	8	rh	<p><b>Voltage Droop Timer Overflow Status</b></p> <p>This status bit indicates that the Voltage Droop timer compare match has happened. If VDTOVEN bit is enabled, then VDTOV can only be cleared by explicitly via VDTOVCLR bit. If VDTOVEN bit is disabled, VDTOV is cleared on a taken Voltage Droop Request (A new Voltage Droop request is taken only if both LJT &amp; VDT are not currently running and no active Voltage Droop request is being processed). VDTOV being set will lead to an interrupt if VDTOVIEN is enabled. Incase SDVOK is set by EVRC before VDT compare match, VDTOV bit is not set.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> Voltage Droop Timer compare overflow has not happened.</li> <li>1<sub>B</sub> Voltage Droop Timer compare overflow has happened.</li> </ul>
<b>VDTOVCLR</b>	12	w	<p><b>Voltage Droop Timer Overflow Status Clear</b></p> <p>This bit clears VDTOV status bit if VDTOVEN bit is enabled. If VDTOVEN bit is disabled, this bit has no effect. This bit always reads as 0.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> This clear bit has no effect on Voltage Droop Timer overflow flag.</li> <li>1<sub>B</sub> Voltage Droop Timer overflow flag is cleared.</li> </ul>
<b>VDTCNT</b>	25:16	rh	<p><b>Voltage Droop Timer Value</b></p> <p>This bit field reflects the current Voltage Droop timer value. VDTCNT value is cleared on timer overflow and on a taken Voltage Droop Request. X us is the compare value. LSB = 1 us. Total range = 65.5 ms</p>

## Power Management System for Low-End (PMSLE)

Field	Bits	Type	Description
0	3:2, 7:6, 11:9, 15:13, 31:26	r	<b>Reserved</b> Read as 0; should be written with 0.

### 12.4 IO Interfaces

The following table defines the signals connecting the PMSLE to other modules and the outside world.

Note that not all signals may be used in all members of the family. Consult the product specific appendix to see the available connections.

**Table 472 List of PMS Interface Signals**

Interface Signals	I/O	Description
HWCFG1IN	in	<b>HWCFG1 pin input</b> Hardware configuration 1 input for activation of EVR33 regulator.
HWCFG2IN	in	<b>HWCFG2 pin input</b> Hardware configuration 2 input for activation of EVRC regulator.
HWCFG4IN	in	<b>HWCFG4 pin input</b> Hardware configuration 4 input for test purposes.
HWCFG5IN	in	<b>HWCFG5 pin input</b> Hardware configuration 5 input for test purposes.
HWCFG6IN	in	<b>HWCFG6 pin input</b> Hardware configuration 6 input for activation of tri-state.
TESTMODEIN	in	<b>TESTMODE pin input</b> Test mode pin input to enable entry into test modes
PORSTIN	in	<b>PORST pin input</b> PORST pin input to trigger warm PORST
PORSTOUT	in	<b>PORST pin output</b> Cold PORST strong pull down control output to drive PORST pin low in case of primary monitor undervoltage events
PORSTWKPD	in	<b>PORST pad weak pull down control output</b> PORST weak pull down control to keep PORST weakly pulled low in case of bond wire breakage. The pull down is inactive to avoid additional current during STANDBY mode
ESR0PORST	out	<b>ESR0 control output during PORST activation</b> Warm PORST signal connected to ESR0 pad to ensure that ESR0 pad is asserted when PORST pin is asserted to propagate the reset
ESR0WKP	in	<b>ESR0 pin input</b> ESR0 pin input for wakeup from STANDBY mode
ESR1WKP	in	<b>ESR1 pin input</b> ESR1 pin input for wakeup from STANDBY mode

## Power Management System for Low-End (PMSLE)

**Table 472 List of PMS Interface Signals (cont'd)**

Interface Signals	I/O	Description
PINAWKP	in	<b>PINA ( P14.1) pin input</b> P14.1 pin input for wakeup from STANDBY mode
PINBWKP	in	<b>PINB (P33.12) pin input</b> P33.12 pin input for wakeup from STANDBY mode
VCAP0	out	<b>DCDC connection to flying capacitor (VCAP0N pin)</b>
VCAP1	out	<b>DCDC connection to flying capacitor (VCAP1P pin)</b>
DCDCSYNC	out	<b>DCDC (P32.4) synchronization output</b>
WUTUFLOW	out	<b>WUT counter underflow signal to CCU6/GTM</b> WUT Underflow output to support WUT calibration
DCDCSYNCGTM	in	<b>DCDC synchronization signal input from GTM</b> Synchronisation input from GTM module to EVRC SMPS regulator
DCDCSYNCCU	in	<b>DCDC synchronization signal input from CCU6</b> Synchronisation input from CCU6 (CCU60 COUT63) module to EVRC SMPS regulator
VDDMLVL	out	<b>VDDM monitor signal to Converter</b> Signal indicating whether VDDM is supplied with 5V or 3.3V. 0: VDDM = 5V 1: VDDM = 3.3V.

## Power Management System for Low-End (PMSLE)

### 12.5 Revision History

#### 12.5.1 Changes from AURIX TC33x PMS V1.0.1 Onwards

**Table 473 Revision History**

Reference	Change to Previous Version	Comment
<b>V1.0.2</b>		
<a href="#">Page 66</a>	Removed description related to external MOSFET complementary switch, since not related to the switched-capacitor DCDC.	
<a href="#">Page 84</a>	Removed VGATE reference from the EVRSTAT.SYNCLCK bit-field description.	
<a href="#">Page 157</a>	Removed the VGATE1P reference from the PMSWCR5.TRISTREQ bit-field description.	
<a href="#">Page 193</a>	Changed VGATE1P and VGATE1N to VCAP0/VCAP1 in the list of PMS interface signals.	
<a href="#">Page 17</a>	Enabled text description in <a href="#">Section 12.2.1.2.5</a> for the user manual.	
<a href="#">Page 22</a>	Aligned switching frequency values mentioned in <a href="#">Section 12.2.2.2</a> to 1.85 MHz (instead of e.g. 1.851 MHz).	
<a href="#">Page 25</a>	Specified register name for the SDFREQSPRD bit-field.	
<a href="#">Page 29</a>	Typo: corrected GPTM to GTM.	
<a href="#">Page 149</a>	Updated description of PMSWCR0.STBYRAMSEL bit-field: 100 <sub>B</sub> and 111 <sub>B</sub> values are reserved (no CPU1 dLMU available).	
<a href="#">Page 162</a>	Updated description of PMSWSTAT2.STBYRAM bit-field: 100 <sub>B</sub> and 111 <sub>B</sub> values are reserved (no CPU1 dLMU available).	
<a href="#">Page 187</a>	Updated the PMTRCSR0.LPSLPEN bit-field description, in the case of PMSLE it is reserved, since the SC-DCDC has no Low-Power Mode (LPM).	
<a href="#">Page 137</a>	Updated signal names in the description of the EVRSDCTRL9.SHHVEN and EVRSDCTRL9.SHLVEN bit-fields.	
<a href="#">Page 124</a>	Updated signal names in the description of the EVRSDCTRL1.TON and EVRSDCTRL1.TOFF bit-fields.	
<a href="#">Page 129</a>	Updated signal names in the description of the EVRSDCTRL4.SDVOKLVL and EVRSDCTRL4.SDLUT bit-fields.	
<a href="#">Page 130</a>	Updated signal name in the description of the EVRSDCTRL5.STCONLIMINC bit-field.	
<a href="#">Page 48</a>	In <a href="#">Table 390</a> , EVRCDCTRL is replaced with CONDUCTANCE.	
<a href="#">Page 172</a>	Corrected the OCDS trigger set in OTSC1.B0EC and OTSC1.B1EC for the 1 <sub>H</sub> value: replaced EVRCDPWM with CONDUCTANCE.	
<a href="#">Page 123</a>	Corrected the EVRSDCTRL0.UP reset value from 0x0 to 0x1.	
<a href="#">Page 128</a>	Corrected the EVRSDCTRL3.BYSEL reset value from 0x01 to 0x00.	
<a href="#">Page 129</a>	Corrected the EVRSDCTRL4.SDLUT reset value from 0x00 to 0x22.	
<a href="#">Page 129</a>	Corrected the EVRSDCTRL4.ZEROBIN reset value from 0x0 to 0x2.	
<a href="#">Page 130</a>	Corrected the EVRSDCTRL5.STCONDEC reset value from 0x1 to 0x3.	
<a href="#">Page 130</a>	Corrected the EVRSDCTRL5.R21 reset value from 0x1b to 0x00.	

## Power Management System for Low-End (PMSLE)

**Table 473 Revision History (cont'd)**

Reference	Change to Previous Version	Comment
<a href="#">Page 143</a>	Corrected the EVRSDCOEFF2.COEFF1 reset value from 0x343 to 0x3d9.	
<a href="#">Page 143</a>	Corrected the EVRSDCOEFF2.COEFF3V3 reset value from 0x2 to 0x4.	
<a href="#">Page 143</a>	Corrected the EVRSDCOEFF2.COEFF5V reset value from 0x4 to 0x2.	
<a href="#">Page 143</a>	Corrected the EVRSDCOEFF2.LINEPOSTHR reset value from 0x1 to 0x4.	
<a href="#">Page 143</a>	Corrected the EVRSDCOEFF2.LINENEGTHR reset value from 0x7 to 0x4.	
<a href="#">Page 145</a>	Corrected the EVRSDCOEFF3.BYPONTHR reset value from 0x00 to 0xa3.	
<a href="#">Page 145</a>	Corrected the EVRSDCOEFF3.SDDIGIN1 reset value from 0x1c to 0x00.	
<a href="#">Page 145</a>	Corrected the EVRSDCOEFF3.SDDIGIN2 reset value from 0x01 to 0x3c.	
<a href="#">Page 90</a>	Updated note on Reset Values of EVRRSTCON register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 95</a>	Updated note on Reset Values of EVRTRIM register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 100</a>	Updated note on Reset Values of EVRMONCTRL register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 105</a>	Updated note on Reset Values of EVRMONFILT register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 110</a>	Updated note on Reset Values of EVRUVMON register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 113</a>	Updated note on Reset Values of EVRUVMON2 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 111</a>	Updated note on Reset Values of EVROVMON register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 114</a>	Updated note on Reset Values of EVROVMON2 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 115</a>	Updated note on Reset Values of HSMUVMON register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 117</a>	Updated note on Reset Values of HSMOVMON register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 121</a>	Updated note on Reset Values of EVROSCCTRL register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 123</a>	Updated note on Reset Values of EVRSDCTRL0 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 124</a>	Updated note on Reset Values of EVRSDCTRL1 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 126</a>	Updated note on Reset Values of EVRSDCTRL2 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 128</a>	Updated note on Reset Values of EVRSDCTRL3 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 129</a>	Updated note on Reset Values of EVRSDCTRL4 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	

## Power Management System for Low-End (PMSLE)

**Table 473 Revision History (cont'd)**

Reference	Change to Previous Version	Comment
<a href="#">Page 130</a>	Updated note on Reset Values of EVRSDCTRL5 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 133</a>	Updated note on Reset Values of EVRSDCTRL6 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 134</a>	Updated note on Reset Values of EVRSDCTRL7 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 136</a>	Updated note on Reset Values of EVRSDCTRL8 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 137</a>	Updated note on Reset Values of EVRSDCTRL9 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 138</a>	Updated note on Reset Values of EVRSDCTRL10 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 140</a>	Updated note on Reset Values of EVRSDCOEFF0 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 141</a>	Updated note on Reset Values of EVRSDCOEFF1 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 143</a>	Updated note on Reset Values of EVRSDCOEFF2 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 145</a>	Updated note on Reset Values of EVRSDCOEFF3 register: The Reset Value is updated by SSW only after cold power-on reset and not after warm resets.	
<a href="#">Page 3</a>	Added a description of the pull-up and pull-down device resistance range for HWCFG[6] and HWCFG[1,2] in order to recognize the high respectively low state setting.	
<a href="#">Page 26</a>	Added a statement that toggling of the EVRSTAT.EVRCMOD lower bit is expected for load currents around the IMAXSC threshold.	
<a href="#">Page 66</a>	Corrected the name of the SMUEN register, from CTRL.SMUEN into CMD_STDBY.SMUEN.	
<a href="#">Page 69</a>	Corrected the name of the SMUEN register, from CTRL.SMUEN into CMD_STDBY.SMUEN.	
<a href="#">Page 28</a>	Removed VDD feedback indication from <a href="#">Figure 135</a> and added statement that no dedicated VDD feedback ball is available in BGA packages.	
<a href="#">Page 123</a>	Corrected reset value of the <b>EVRSDCTRL0.LCK</b> bit from 0 to 1.	
<a href="#">Page 124</a>	Corrected reset value of the <b>EVRSDCTRL1.LCK</b> bit from 0 to 1.	
<a href="#">Page 126</a>	Corrected reset value of the <b>EVRSDCTRL2.LCK</b> bit from 0 to 1.	
<a href="#">Page 128</a>	Corrected reset value of the <b>EVRSDCTRL3.LCK</b> bit from 0 to 1.	
<a href="#">Page 129</a>	Corrected reset value of the <b>EVRSDCTRL4.LCK</b> bit from 0 to 1.	
<a href="#">Page 130</a>	Corrected reset value of the <b>EVRSDCTRL5.LCK</b> bit from 0 to 1.	
<a href="#">Page 133</a>	Corrected reset value of the <b>EVRSDCTRL6.LCK</b> bit from 0 to 1.	
<a href="#">Page 134</a>	Corrected reset value of the <b>EVRSDCTRL7.LCK</b> bit from 0 to 1.	
<a href="#">Page 136</a>	Corrected reset value of the <b>EVRSDCTRL8.LCK</b> bit from 0 to 1.	
<a href="#">Page 137</a>	Corrected reset value of the <b>EVRSDCTRL9.LCK</b> bit from 0 to 1.	

## Power Management System for Low-End (PMSLE)

**Table 473 Revision History (cont'd)**

Reference	Change to Previous Version	Comment
<a href="#">Page 138</a>	Corrected reset value of the <b>EVRSDCTRL10.LCK</b> bit from 0 to 1.	
<a href="#">Page 140</a>	Corrected reset value of the <b>EVRSDCOEFF0.LCK</b> bit from 0 to 1.	
<a href="#">Page 141</a>	Corrected reset value of the <b>EVRSDCOEFF1.LCK</b> bit from 0 to 1.	
<a href="#">Page 143</a>	Corrected reset value of the <b>EVRSDCOEFF2.LCK</b> bit from 0 to 1.	
<a href="#">Page 145</a>	Corrected reset value of the <b>EVRSDCOEFF3.LCK</b> bit from 0 to 1.	
<b>V1.0.3</b>		
<a href="#">Page 4</a>	Updated <a href="#">Figure 122</a> with note about xQFP80 and xQFP100 packages.	
<a href="#">Page 37</a>	Enabled description for activating EVR33 short detection scheme.	
<a href="#">Page 119</a>	Enabled description of <b>EVR33CON</b> register for EVR33 short detection configuration.	
<a href="#">Page 130</a>	Updated <b>EVRSDCTRL5.STTH32ROFF</b> description by removing some empty placeholders.	
<a href="#">Page 136</a>	Updated <b>EVRSDCTRL8.VINDIG</b> description by removing some confusing questionmarks.	
<a href="#">Page 58</a>	Removed question marks and updated PMSTAT0.CPU[y] & LS status bit values for the “System during Sleep Mode” condition.	
<a href="#">Page 177</a>	Updated description of <b>PMCSR0</b> register (for CPU0).	
<a href="#">Page 178</a>	Updated description of <b>PMCSR1</b> register, indicating that this register has no function if CPU1 is not available on a product variant.	
<a href="#">Page 179</a>	Updated description of <b>PMCSR2</b> register, indicating that this register has no function if CPU2 is not available on a product variant.	
<a href="#">Page 180</a>	Updated description of <b>PMCSR3</b> register, indicating that this register has no function if CPU3 is not available on a product variant.	
<a href="#">Page 181</a>	Updated description of <b>PMCSR4</b> register, indicating that this register has no function if CPU4 is not available on a product variant.	
<a href="#">Page 182</a>	Updated description of <b>PMCSR5</b> register, indicating that this register has no function if CPU5 is not available on a product variant.	
<b>V1.0.4</b>		
<a href="#">Page 70</a>	Corrected the blanking filter minimum time: reduced from 1ms to 40us. There is no issue if a longer filter time has been configured (e.g. 1ms), but the minimum required duration is of only 40us.	
<a href="#">Page 43</a>	Corrected in <a href="#">Figure 144</a> the VDDPD secondary monitoring over-voltage and under-voltage levels.	
<a href="#">Page 3</a>	Removed reference to EVR33 disabling bit.	
<a href="#">Page 33</a>	Removed reference to EVR33 disabling bit.	
<a href="#">Page 3</a>	Added information about VEXT-buffered PU1 pads state after standby mode entry (regardless of the HWCFG[6] setting).	
<a href="#">Page 66</a>	Added information about VEXT-buffered PU1 pads state after standby mode entry (regardless of the PMSWCR5.TRISTEQ setting).	
<a href="#">Page 69</a>	Added information about VEXT-buffered PU1 pads state after standby mode entry (regardless of the PMSWCR5.TRISTEQ setting).	

---

**Power Management System for Low-End (PMSLE)**
**Table 473 Revision History (cont'd)**

<b>Reference</b>	<b>Change to Previous Version</b>	<b>Comment</b>
<a href="#">Page 70</a>	Added information about VEXT-buffered PU1 pads state after standby mode entry (regardless of the PMSWCR5.TRISTEQ setting).	
<a href="#">Page 8</a>	Corrected VDDPD voltage in <a href="#">Figure 123</a> .	
<a href="#">Page 6</a>	Corrected VDDPD voltage range in <a href="#">Table 382</a> .	
<a href="#">Page 6</a>	Corrected VDDPD voltage range in <a href="#">Table 383</a> .	
<b>V1.0.5</b>		
<a href="#">Page 50</a>	Added statement to the Standby Mode (Only VEVRSB supplied).	
<a href="#">Page 59</a>	Added statement about Standby Entry trigger.	
<a href="#">Page 60</a>	Added statement about Standby Entry trigger.	
<a href="#">Page 66</a>	Added statement on Standby Entry trigger event and a standby entry trigger to the steps to enter standby.	
<a href="#">Page 66</a>	Added bullet list item.	
<a href="#">Page 69</a>	Added bullet list item.	
<a href="#">Page 50</a>	Changed bullet list item.	
<a href="#">Page 62</a>	Replaced sentence in <a href="#">Chapter 12.2.3.4.6</a> .	

## Memory Test Unit (MTU)

### 13 Memory Test Unit (MTU)

The Memory Test Unit (MTU) controls and monitors the test, initialization and data integrity checking functions of the various internal memories in the device.

Each SRAM in this Platform has some digital logic surrounding it, known as SRAM Support Hardware (SSH). An SSH is a hardware block which controls the Error Detection & Correction, Memory Built-In-Self-Test(MBIST) of internal memories. Each SSH block provides an unified interface for controlling its various functionalities. There are multiple such SSH instances, each of which controls one or more of the different internal memories. The Memory Test Unit (MTU) in the AurixPlus Platform provides register interfaces to configure and control these various memory controllers.

#### 13.1 Feature List

The main features of the MTU are as follows:

- Unified interface to internal SSH instances
  - The MTU provides an unified register interface to control the operation and functionality of each internal SSH instance.
  - Various configurable test types for each of the SRAM blocks in the system can be controlled via the MTU.
- Data Initialization
  - Each SRAM block in the system can be hardware initialized via the MTU.
  - Security sensitive memories can be autoinitialized to prevent data read-out via the MTU.
- Memory Error Correction & Detection
  - Memory error detection/correction for the SRAM blocks in the system can be configured via the MTU.
  - Correctable and uncorrectable error detection.
  - Address Error detection
- Alarm notification to SMU: From each SRAM/SSH, 3 alarms are sent to the MTU, which are then forwarded to the SMU. These are the CE alarm, UCE alarm and ME alarm.

#### 13.2 Overview

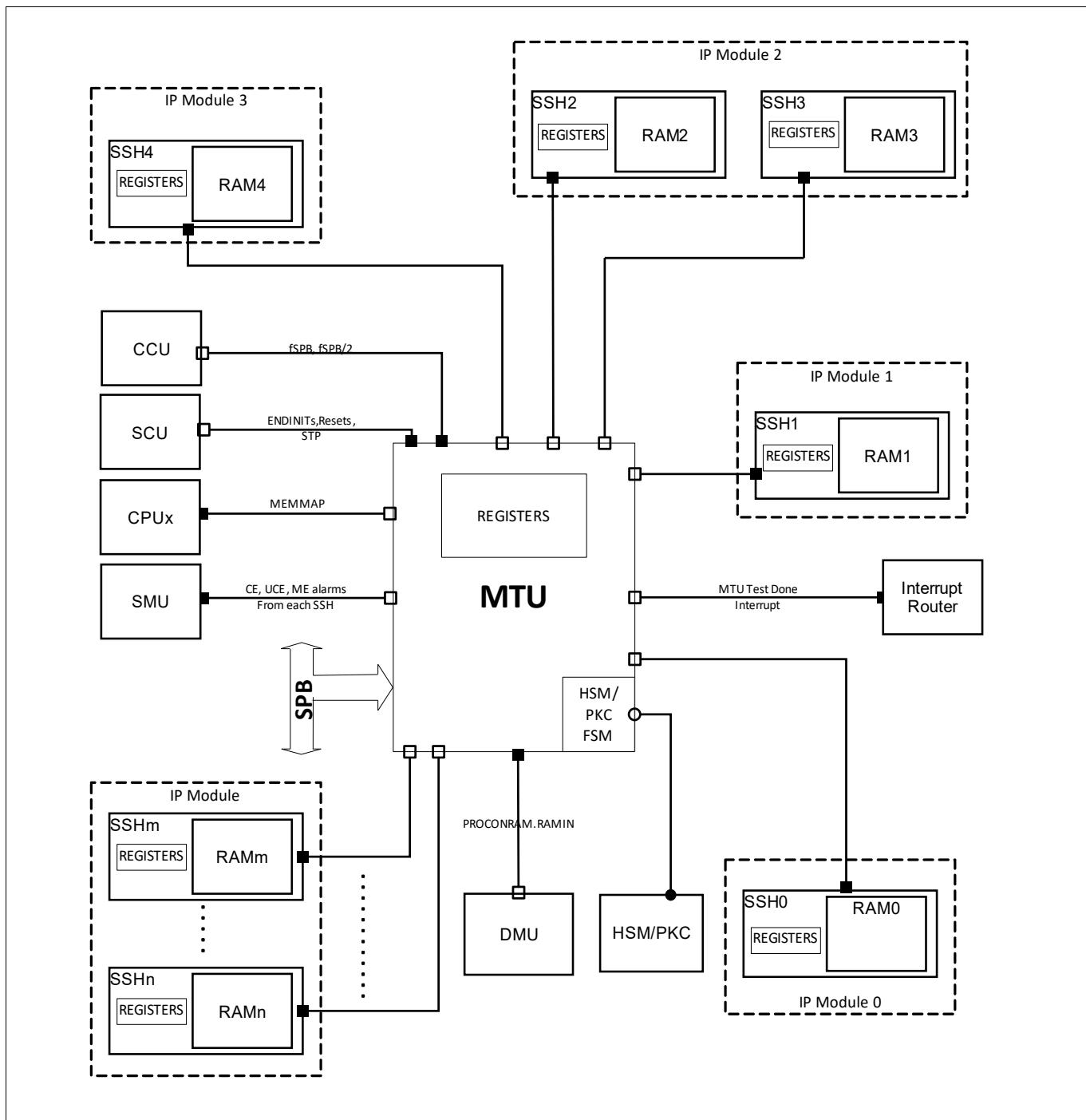
[Figure 153 “System View: MTU, SSHs and SRAMs in IPs” on Page 2](#) shows the system level view of the MTU, SRAM Support Hardware (SSH) Instances, and SRAMs in the system.

Different IP Modules in the system (E.g. CPU, LMU, CAN etc) may have one or more than one SRAM inside them. Surrounding every SRAM is its own SSH module, with its own registers ([SRAM Support Hardware \(SSH\) Registers](#)) and MBIST logic.

Each SSH is separately connected to the MTU via internal interfaces

The MTU itself is connected to the SPB bus, and each SSH's registers can be accessed via this SPB interface. Each SPB access is internally forwarded (and returned) to (and from) the corresponding SSH by the MTU.

## Memory Test Unit (MTU)



**Figure 153 System View: MTU, SSHs and SRAMs in IPs**

### 13.3 Functional Description

#### 13.3.1 Major Functional Changes from TC39xA-Step to TC39XB-Step / TC38XA-Step

There are several major functional changes compared to the TC39xA-step, which may have an impact on the software/application. Only the most important ones are highlighted here.

- The SRAM error notifications are now consolidated to 3 alarms from each SSH- CE alarm, UCE alarm and ME alarm.

## Memory Test Unit (MTU)

- Entering Test mode (i.e. setting MTU\_MEMTEST.MEMxEN = 1), or starting an MBIST test will trigger an UCE alarm.
- Similarly disabling any of the alarm sources or safety mechanisms will also trigger an SSH ME alarm.
- Software has to take care to handle these alarms generated while triggering these operations.
- The alarm status bits in the **MCI\_ECCD (i=0-95)** register are named CERR, UCERR and MERR. After any alarm, the software shall clear the corresponding bits after an alarm to get further alarms.
- New registers **MCI\_ALMSRCS (i=0-95)** and **MCI\_FAULTSTS (i=0-95)** have been added to enable and get the status of errors mapped to UCE and ME alarm.
- New bits PERMERRx have been added in the **MCI\_ECCD (i=0-95)**.
- It is made possible for the Non destructive test to be run with configurable march elements and directions.
- SRAM\_CLR is now defined to use ECC-correct zero data for the initialization (on the A-step, the actual data used for the initialization depended on the module).
- The reset domains of SSH are redefined.
- Alarm status bits in ETRRx, ERRINFOx and FAULTSTS registers are preserved until a power on reset. Similarly additional reset domains are defined within the SSH.

### 13.3.2 SRAM Support Hardware (SSH)

Each internal RAM (or multiple RAMs, depending on the configuration) in the AurixTC3XX Platform which is to be internally tested has some additional logic attached to it. This is the SRAM support hardware or SSH. All the SSH instances for the different memories in the device are connected to an internal bus via an unified MTU-SSH Interface. The SSH provides a direct access to the memories, without involving the CPU. Even small memories that are not directly accessible via the CPU can be tested via the SSH.

### 13.3.3 Control and Status Interfaces

The following section explains the hardware interface to access each individual SSH instance by the CPU via the peripheral bus.

#### 13.3.3.1 Interface to the CPU

The registers of an individual SSH can be accessed by the CPU via the MTU over the peripheral bus.

The registers of individual controllers are located in the SSH block. Logically they all have an MTU system address and can be accessed through normal 16 bit SPB bus accesses via the MTU, but since these have to go via the MTU->SSH interface, the accesses to individual SSH registers will be slower (maximum 20 SPB clock cycles for SSHs running synchronous to SPB clock) than normal SPB accesses to peripheral registers. Via the MTU, the SSH registers shall be accessed with 16 bits wide accesses (both Read and Write).

In general, if any SSH register is accessed while a test is on-going, then for writes: the values are not written to the register, and read access return the correct value. In both cases the access finishes cleanly on the SPB and there is no bus error. The **MCI\_MCONTROL (i=0-95)** register can be written to during a test to clear the start & resume bits. But in case other bits of MCONTROL are modified after a test has started (i.e. after MCONTROL.START has been set to 1) - then this may result in unpredictable behaviour.

It is forbidden to change any test parameter in the registers (i.e. CONFIG0, CONFIG1 and MCONTROL (except START & RESUME)) once a test has started. If the SSH registers are changed during a test (i.e. before the done is set) -> this may result in undefined and erroneous behaviour.

## Memory Test Unit (MTU)

### 13.3.4 Enabling the SRAM Support Hardware (SSH)

Each SSH in the AurixPlus Platform has an enable bit associated with it in the MTU\_MEMTEST $x$  ( $x = 0 - 2$ ) register.

Please refer to the **MEMTEST $i$  ( $i=0-2$ )** registers for the SSH enable bits implemented in the AurixPlus Platform.

The control registers of each SSH are accessible only when the SSH is enabled (MTU\_MEMTEST.MEM $x$ \_EN = 1).

The only exceptions to this are registers **MCI\_ECCS ( $i=0-95$ )**, **MCI\_ECCD ( $i=0-95$ )** and **MCI\_ETRR $x$  ( $i=0-95; x=0-4$ )**, **MCI\_ERRINFO $x$  ( $i=0-95; x=0-4$ )**, **MCI\_ALMSRCS ( $i=0-95$ )** and **MCIFAULTSTS ( $i=0-95$ )** (in each SSH instance) which are available at all times to permit easy runtime access to ECC features, i.e., these particular registers (i.e. ECCS, ECCD, ETRR $x$ , ERRINFO $x$ , ALMSRCS and FAULTSTS) can be read and written with the only pre-condition that the MTU clock is enabled via the MTU\_CLC register.

All other registers in the SSH can be accessed only when the SSH is enabled via the MEMTEST register, otherwise the MTU returns an SPB error.

For certain SSHs, enabling or disabling the SSH via the MEMTEST register may result in whole or part of the memory to be automatically initialized depending on the configuration (see "["Security-Sensitive Memories and AutoInitialization" on Page 4](#)"). This auto-initialization can take hundreds of clock cycles to complete. During this time, no SSH register is accessible (not even the registers which normally do not need the SSH to be enabled, to be accessed) - and any register access to the SSH running the auto-initialization will result in an SPB bus error.

Note that the clock input to the module containing the SRAM has to be enabled via the CCU registers, before any SSH registers can be accessed via the MTU. Other than this CCU clock configuration, there is no need to enable the module clock (e.g. using the CLC register) to access SSH registers.

**Note:**

2. When an SSH is enabled, functional access to the memory is temporarily unavailable. . When a memory is being tested, the software should prevent an attempted functional access to that memory (e.g. while testing a CPU's local memories, the CPU should be in an idle state, or executing from other memory).
3. Correct execution of an SSH operation (e.g. an MBIST test) requires that both the MTU and the module containing the memory-under-test are operational for the duration of the SSH operation(i.e. test). A reset of the module containing the memory-under-test, when an SSH is enabled can possibly result in unexpected behaviour. Software has to take care that events such as application reset, module resets, standby entry, clock frequency change etc. are not triggered when an SSH is enabled. For example - a module reset may result in the module trying to access the SRAM functionally after the reset, when the SSH is still enabled (and thus functional access is disabled)- such scenarios have to be avoided by software.
4. If communication is on-going (e.g. SSH register read or write) between the MTU and SSH when a module reset happens - the communication is aborted and a bus error triggered on the SPB immediately within a few cycles, without any timeout. Only the on-going read/write may be corrupted and future communication is not affected.
5. In case invalid SSH registers are accessed (e.g. registers not existing in an SSH instance, or a non existing SSH instance), this will result in a bus error on the SPB.

**NOTE:**

#### 13.3.4.1 Security-Sensitive Memories and AutoInitialization

In the AurixTC3XX Platform, certain internal memories are considered security sensitive. This means that during certain applications which are security sensitive, reading or modifying the contents of these memories via the SSH modules has to be prevented by the MTU. For these memories, enabling the corresponding SSH results in the memory being cleared, before the SSH is actually enabled. This operation can take hundreds of clock cycles.

This applies to the following memories (The corresponding enable bits in the MTU\_MEMTEST $x$  register have \_EN appended to the memory name):

## Memory Test Unit (MTU)

- CPUx\_DMEM(x = 0 - 5) - Only the cache part of the memory is considered security sensitive.
- CPUx\_DTAG(x = 0 - 5)
- CPUx\_PMEM(x = 0 - 5) - Only the cache part of the memory is considered security sensitive.
- CPUx\_PTAG(x = 0 - 5)

In the case of non-security applications (e.g. safety), it may be important that the modification of the memory contents is allowed via the SSH (e.g. for ECC error injection or run-time self-test).

It is possible to enable or disable this auto-data-init and partial-erase via PROCONRAM register in the DMU.

Please note that the auto-data-init and partial-erase are nominal functions, and it is not intended to guarantee that they satisfy the security requirements under all conditions - i.e. it may be possible to interrupt the initialization.

Note: The security of memory initialization during startup is not affected.

### 13.3.4.1.1 Security Applications

For security applications, the contents of the security sensitive memories can be autoinitialized to erase the existing contents. This is controlled by the PROCONRAM register in the DMU (please refer the DMU chapter).

If PROCONRAM.RAMIN = 00, 01 or 10 then automatic memory content initialization is enabled. PROCONRAM configures whether the initialization is triggered by cold resets, warm resets or both.

In these modes, an automatic initialization of security-sensitive memories is also triggered whenever the corresponding MTU\_MEMTEST.MEMx\_EN or MTU\_MEMMAP is changed (i.e. The corresponding SSH is enabled or disabled or if memory map mode is changed).

The **MEMSTAT*i*** (*i*=0-2) register bits indicate whether an automatic data initialization of Memory *x* has been triggered by a change of state of MEMTEST.MEMx\_EN or MTU\_MEMMAP and the initialization sequence has not yet completed. If a MEMx\_AIU bit in a MEMSTAT register is set, this means that Autoinitialization for that memory is still underway. The SSH is enabled only after, and disabled before the Auto-initialization starts. The software can wait for the Auto-initialization to be completed by polling this bit.

### 13.3.4.1.2 Non-Security Applications

If PROCONRAM.RAMIN=11 then no automatic initialization of RAM content is performed on a reset and no automatic initialization of RAM content is performed when SSH modules are enabled or disabled with MTU\_MEMTEST.MEMx\_EN or MTU\_MEMMAP register bits.

This permits the use of the SSH by an application (E.g. for error injection, data modification or runtime memory testing) without unwanted corruption of memory content.

### 13.3.4.2 Memory Map selection

The Memory Mapping Enable register (refer **MEMMAP**) has configurable control bits to map the CPU caches and tags to the system address space. For the address spaces from which the caches & tag memories can be accessed - kindly refer to the Memory Maps chapter.

The MBIST tests and other SSH operations work on the SRAM independent of the MEMMAP register settings.

### 13.3.5 SRAM Support Hardware (SSH) Operation

The operation and functionality of each internal SSH instance can be controlled via a set of registers. These registers are accessible via the MTU. As explained in the section "**Enabling the SRAM Support Hardware (SSH)" on Page 4**", an SSH instance has to be enabled first, before the memory can be tested.

The main functionalities provided by each SSH instance are explained in the following sections.

## Memory Test Unit (MTU)

### 13.3.5.1 Memory Testing and Initialization

The following section explains how to configure the SSH instances via the MTU, to perform various tests on the memory and obtain the results, or initialize the memories via the SSH.

**Note:** *The following operations can be performed only when the SSH instance under test has been enabled using the corresponding MEMx\_EN bit in the MTU\_MEMTESTx (x = 0-2) register.*

#### 13.3.5.1.1 Starting a Memory Test Sequence

Each memory test sequence is started by writing to the MCONTROL.START bit of the corresponding SRAM Support Hardware (Refer [MCi\\_MCONTROL \(i=0-95\)](#) register). When the test is complete, the MSTATUS.DONE is set by the hardware when the software has cleared the MCONTROL.START bit. Software can poll the [MEMDONEi \(i=0-2\)](#) registers in the MTU to get the status of test completion. These registers just reflect the DONE bit in the [MCi\\_MSTATUS \(i=0-95\)](#) register of each SSH.

Before the MCONTROL.START is set, the software should properly configure the Configuration registers, CONFIG0 and CONFIG1 registers. (Refer [MCi\\_CONFIG0 \(i=0-95\)](#), [MCi\\_CONFIG1 \(i=0-95\)](#)).

CONFIG0.ACCTYPE specifies the access type (Read or Write) to be performed on each single address in the current marching element, while CONFIG1.ACSPAT specifies the access pattern. CONFIG0.NUMACCS specifies the total number of accesses to a single address in the current marching element. The SSH supports some complex addressing schemes. The software can enable this using the CONFIG1.AG\_MOD bits.

Once the test is complete (MSTATUS.DONE=1), the MSTATUS.FAIL bit will be set in case of any failures.

When a Non-destructive test is configured using the CONFIG1.AG\_MOD bits, the MSTATUS.FAIL bit is not set - instead, the software has to check the ECCD, ETRR and ERRINFO registers for any errors detected during the test. Complex algorithm may require several march starts to get a full test.

#### 13.3.5.1.2 Memory Test Done Interrupt

The MTU provides an interrupt to the interrupt router (IR). The interrupt signifies the completion of all running tests. The reset value of this signal is high. When a test on any SRAM is started, this signal goes to low. On completion of all on-going tests, this signal again goes high, and this rising edge triggers the interrupt.

#### 13.3.5.1.3 Getting Detailed Memory Test Results

The MSTATUS.FAIL and MSTATUS.DONE (can be polled from the MTU itself via [MEMDONEi \(i=0-2\)](#) register) bits provide a general pass/fail information and test completion status.

If MCONTROL.FAILDMP = '1', the test stops after a failure and the fail information is immediately available for dump. MSTATUS.FDA (fail dump available) is set in this case. Any dump information has to be polled from registers RDBFL and ETRR(0). RDBFL contains the fail bit map and ETRR the failed address. Reading MSTATUS and the RDBFL(n-1) registers with MSTATUS.FDA = '1' will reset MSTATUS.FDA back to '0'. A consequent setting of MCONTROL.RESUME will resume the interrupted test sequence.

The RANGE register can be used to run consecutive tests on constantly shifted memory ranges so that in the end the complete memory has been analyzed.

Please note that if a test is stopped due to an intermediate failure, the MSTATUS.DONE is not set. MSTATUS.DONE is set only once the whole test sequence is completed and when MCONTROL.START has been cleared by software.

#### Error Injection During Memory Tests

It is possible to manually inject errors during memory tests (for example, to test the software).

For a non-destructive test, an error in the data can be introduced by programming a word with wrong ecc before the test, via the ECCMAP bits.

## Memory Test Unit (MTU)

During a destructive march test, it is possible to inject data errors via the RANGE.INJERR bit. When this bit is set (and RANGE.RAEN = 1), then the RANGE.ADDR field is taken as a pointer to a physical SRAM location, to which write accesses during the test are not executed. Software can then write a particular value (i.e. using single SRAM write access using the RDBFL register) before a march test, and then run the test with RANGE.INJERR and RAEN = 1.

The test then runs over the full memory, and on the address corresponding to RANGE.ADDR, writes are not executed. This results in mismatch of the expected data during the test, resulting in a FAIL.

With errors injected, all normal diagnostics and notifications can be tested- that is, alarms are triggered, errors are tracked in the ETRR/ERRINFO during a non-destructive test. And during a march test, FAIL bit is set, and when FAILDUMP = 1, the fail bitmap is obtained and FDA is set.

The fail bitmap is simply (Expected data pattern) XOR (Actual data pattern).

Address errors are triggered during the test by simply setting SFLE bit to 1. Note that this will trigger an address error from each address, and during a non-destructive test, result in the ETRR/ERRINFO getting filled with address errors.

### 13.3.5.1.4 Filling a Memory with Defined Contents

The SSH can be used to fill a memory range or a complete memory with a defined pattern very fast, i.e. one write access per cycle with the full memory data width, using the MCONTROL.DINIT bit. For this, it has to be ensured that MCONTROL.SRAM\_CLR = 0.

Before setting the MCONTROL.DINIT, the software should first fill the RDBFL register with the desired bit pattern (please refer to the [MCI\\_RDBFLy \(i=0-95;y=0-66\)](#)). It is not mandatory to have a valid ECC code in this pattern.

Next, the RANGE register needs to be set with the memory range into which the pattern needs to be filled.

Next, the MCONTROL.DINIT needs to be set, and then the MCONTROL.START bit should be set to start initializing the RAM. The software should then clear the MCONTROL.START. When MSTATUS.DONE bit is set by the hardware, the memory filling operation is complete.

This method of SRAM initialization using the DINIT bit is not supported for certain SRAMs. These exceptions are mentioned in the product specific appendix chapter.

### 13.3.5.1.5 Initializing SRAMs

Using the MCONTROL.SRAM\_CLR bit, it is possible to initialize the complete SRAM. This is supported for all SSHs and SRAMs.

For this operation, enable the SSH (MEMTESTx.MEMxEN = 1), and:

(Note that this will trigger an UCE alarm. Therefore, the alarm reaction may need to be disabled before. Software can set ALMSRCS.OPENE before)

1. Set the MCONTROL.SRAM\_CLR.
2. Start the initialization using MCONTROL.START.
3. Wait for MSTATUS.DONE (can be also polled in the MTU) to be reset and clear the MCONTROL.START.
4. Wait for the end of the initialization by polling the MSTATUS.DONE (can be polled via MTU\_MEMDONE register also) bit.
5. Clear the MCONTROL.SRAM\_CLR and leave the test mode (MEMTESTx.MEMxEN = 0).

With this initialization, the complete SRAM will be filled with ECC-correct zero value.

Clear the UCERR and OPERR flags set due to this operation. Re-enable ALMSRCS.OPENE if it was disabled before the test.

---

## Memory Test Unit (MTU)

Note that for SRAMs described in [Chapter 13.3.4.1](#), enabling or disabling the SSH via MEMTESTx register takes time, for the initialization to complete. Therefore software must wait for the corresponding MEMSTATx.AIUX bits to be cleared, to ensure that this operation is complete.

## Memory Test Unit (MTU)

### 13.3.5.1.6 Reading a Single Memory Location

The SSH can also be used to read the contents of a single word. The RDBFL register holds the contents of a complete memory word and thus it is possible to read all memory bits, including ECC or parity bits. The necessary steps are:

(Note that this will trigger an UCE alarm. Therefore, the alarm reaction may need to be disabled before)

1. Enter memory test mode
2. Initialize registers  
RANGE := RAEN = 0 (range disabled = single address) & address to be read  
CONFIG0 := 1001H (NUMACCS = 1<sub>H</sub>, ACCSTYPE = 01<sub>H</sub> (read))  
CONFIG1 := 0000H (linear mode, non inverted pattern)
3. MCONTROL := 4009H (FAILDMP = 0, direction up, start): Start read operation. MSTATUS.DONE will be cleared now.
4. MCONTROL := 4008H (clear START)
5. Wait for MSTATUS.DONE to be set again (Poll the corresponding bit in the **MEMDONE<sub>i</sub> (i=0-2)** register).
6. Read RDBFL register
7. Leave memory test mode
8. Clear the UCERR and OPERR flags set due to this operation.

### 13.3.5.1.7 Writing to a Single Memory Location

The SSH can also be used to write the contents of RDBFL register to a single memory location. RDBFL holds the contents of a complete memory word and thus it is possible to write to all memory bits, including ECC or parity bits. The necessary steps are:

(Note that this will trigger the UCE alarm. Therefore, the alarm reaction may need to be disabled before)

1. Enter memory test mode
2. Initialize registers  
RDBFL := write data  
RANGE := RAEN= 0 (range disabled = single address) & address to be written to  
CONFIG0 := 1000H (NUMACCS = 1<sub>H</sub>, ACCSTYPE = 00<sub>H</sub> (write))  
CONFIG1 := 0000H (linear mode, non inverted pattern)
3. MCONTROL := 4009H (FAILDMP = 0, direction up, start): Start write operation. MSTATUS.DONE will be cleared now.
4. MCONTROL := 4008H (clear START)
5. Wait for MSTATUS.DONE to be set again (Poll the corresponding bit in the **MEMDONE<sub>i</sub> (i=0-2)** register).
6. Leave memory test mode
7. Clear the UCERR and OPERR flags set due to this operation.

### 13.3.6 Resets and Clocks in the MTU, SSH & SRAM

Since the MTU is a central module and the different SRAMs (and surrounding SSHs) are embedded in different modules, multiple clocks and resets come into the picture.

#### 13.3.6.1 Clock Domains

The MTU runs on the SPB clock. The interface between the MTU and SSH runs on a clock which is always at a fixed divider of 1/2 times the SPB clock. The SRAM and the SSH logic (i.e. Registers, FSM etc) run at the module clock

## Memory Test Unit (MTU)

frequency (i.e. the module where the SRAM is embedded in). The SSH SFR accesses as well as alarm forwarding still work correctly even if system is running with lower clock divider in LPDIV mode.

**Attention:** Whenever accessing the SSH registers, the clock frequency of the corresponding module containing the SSH/SRAM should be equal to or greater than  $f_{SPB}/10$ . For example, with  $f_{SPB} = 100MHz$ , a module's clock frequency should be atleast 10MHz when reading or writing SSH registers in the module.

**Table 474 MTU, SSH and SRAMs Clock Domains**

	Clock
MTU	$f_{SPB}$ , Communication to SSH on $f_{SPB}/2$ clock
SSH	Communication to MTU on $f_{SPB}/2$ . Other SSH logic on Module Clock
SRAM	Module clock
Alarms	Propagated to SMU on $f_{SPB}/2$ clock

### 13.3.6.2 Reset Domains

There are different reset domains to be considered within the MTU, SSH and SRAMs.

**Table 475 MTU, SSH and SRAMs Reset Domains**

Module/Register/Function	Reset Domain
MTU & Alarm Path MTU<->SMU	Application Reset
MTU: Interface to SSH (and FFs in the interface path)	Application Reset
SSH: FSMs (MBIST FSM & Communication with MTU)	Application Reset
SSH: ECCS, ALMSRCS Registers- Notification Enable Bits	Application Reset
SSH: Alarm status Flags (ECCD.CERR, UCERR, MERR)	Application Reset
SSH: ETRR/ERRINFO, FAULTSTS Registers, ECCD.VAL, ECCD.PERMERR and ECCD.EOF bits	Warm PORST
SSH: Test Related Registers (MCONTROL, CONFIG0/1, MSTATUS, RDBFL, RANGE)	Application Reset
SRAM	Cold PORST

Exception for SCR FSI: The SSHs within the SCR are a special case since the SCR resides within the PMS subsystem and has separate reset domains (Asynchronous).

#### 13.3.6.2.1 Alarm Handling after Reset

When an alarm occurs, the system may perform a reset.

The Alarm status bits in the ECCD register (i.e. ECCD.CERR, UCERR and MERR) are cleared after an application reset.

However, the error status bits (i.e. ETRR, ERRINFO and FAULTSTS registers) are still available until a power-on reset for diagnosis purposes - they can only be cleared by software and are reset only with a warm PORST.

The alarm itself is also cleared with an application reset. This prevents a single alarm creating a reset loop.

## Memory Test Unit (MTU)

### 13.3.7 SRAM Addressing and Scrambling

When considering SRAM addressing, different levels of addressing need to be considered. At the highest level, all SRAMs which appear on the overall system memory map could be potentially accessed via a portion of the system address space. This is the logical address space to access the SRAM as far as the system is concerned.

However, from just the SSH point of view, this system level logical address cannot be seen, since the SSH is tightly coupled to the SRAM, and the system address translation or mapping occurs at a higher level before arriving at the SSH. Therefore, unless otherwise specifically mentioned, any “address” in this chapter does not correspond to any of the system address that is mentioned in the memory map chapter.

Within the SSH, the logical addresses increment linearly from 0x00 until a maximum address depending on the size of the SRAM. This maximum address can be inferred from the default (reset) value of the RANGE.ADDR field for each SSH.

During normal system operation , the incoming address is directly input to the SRAM. This address at the input to the SRAM is also stored in ETRR registers in case of any error.

In order to access the error address location stored in the ETRR during normal functional mode, the same address can be provided to the RANGE.ADDR, with RANGE.RAEN = 0 and MCONTROL.EN\_DESCR = 0.

### 13.3.8 MBIST Algorithms

In order to check the integrity of the SRAM and its contents, an MBIST may be run by configuring the SSH.

The test type and parameters are programmed via the SSH registers -CONFIG0 and CONFIG1, and certain parameters via the MCONTROL register.

Additional memory test algorithms are supported by the SSH, but used in production test modes only.

#### 13.3.8.1 Non-Destructive Test (NDT)

The Non-Destructive Test preserves the content of SRAMs exactly as it was before the test. It allows running memory test during application runtime, without destroying any application data in the memory. The following preconditions apply when running the NDT.

1) The SRAM has to be completely initialized with ECC correct data. This may be especially important when running the test after a cold power on reset, when the SRAM contents may not be defined.

For CPU and LMU memories, it may be possible to enable automatic initialization after a reset via firmware using the settings in the PROCOND register. To initialize SRAMs using the MTU, please refer to [Chapter 13.3.5.1.4](#) and [Chapter 13.3.5.1.5](#). In addition, it is also sufficient to initialize SRAMs by writing data to it from the CPU or DMA.

2) It is not possible to access the SRAM functionally when running a test (as long as the SSH is enabled via the MTU\_MEMTEST.MEMxEN bit). Any such access may stall and may result in some undefined state. Software has to take care that for example the CPU PMEM or DMEM SRAMs may be implicitly accessed if caches are enabled.

Therefore, for all CPU SRAMs and LMUs, before entering the test mode (MEMTESTx.MEMxEN = 1) - the program and data caches shall be disabled. It shall be ensured that other masters (e.g. another CPU, DMA or debugger) do not access the SRAM under the test. For peripherals SRAMs, it shall be ensured that the module does not access the SRAM under test.

#### NDT Algorithm

The NDT algorithm reads a word from the SRAM (DATA + ECC bits), inverts it and writes it back. If the ECC does not match the expected value when reading, an error is expected and notified.

## Memory Test Unit (MTU)

This is denoted by the sequence {r, w\*}. Here r denotes a read access, and w\* denotes a write access with inverted data compared to original SRAM contents.

A '\*' symbol in this notation always indicates that the data during that access is inverted with respect to the original content in the RAM.

A write access during the NDT test always inverts the previously read data, before writing it back to the SRAM. This means, to preserve the original contents of the SRAM, a sequence with an even number of writes is required.

A simple example of such a sequence is: {r, w\*, r\*, w}.

Steps (Apply all the steps one after the other on each word, and then move to the next word, until the complete address range is covered):

1. r: Read data word including check bits.
2. w\*: Write back all bits inverted.
3. r\*: Read data word including check bits (All bits are inverted compared to original SRAM contents).
4. w: Write back all bits inverted (The Data is now same as the original SRAM contents).

After this sequence, the user data is undisturbed and every bit would have seen '0' and '1'.

The NDT supports only a linear address sequence.

Note that if an NDT is programmed without a read as the very first access, then any previously read arbitrary data may be used for the write, resulting in data corruption.

## Programming the NDT

To run the Non-Destructive Test, the software has to program the CONFIG0, CONFIG1 and MCONTROL registers, as well as the RANGE register (address range of the RAM to be tested).

**Note:** *By default, after an application reset the RANGE register contains a value which corresponds to the complete memory range. Hence as long as this register is not changed by the software after an application reset, the complete memory is always tested by default (recommended).*

The end of the test is signalled by the DONE bit in the MSTATUS register. Error information can be obtained from the ECCD, ETRR and ERRINFO registers.

**Attention:** *The MSTATUS.FAIL bit (refer [Mci\\_MSTATUS \(i=0-95\)](#)) is set during an NDT only when an address error is detected.*

Similarly the FAILDUMP shall not be set during an NDT, and correspondingly the FDA bit is also irrelevant during an NDT.

The NDT algorithm is selected by programming CONFIG1.AG\_MOD = 0x5. (Refer [Mci\\_CONFIG1 \(i=0-95\)](#) register).

The number of accesses and each corresponding access type (read or write) shall be programmed in the CONFIG0.NUMACCS and ACCSTYPE fields respectively.

The CONFIG1.ACSPAT bits shall be programmed to 1 when the last programmed read access was with inverted data. Here "inverted" is always with respect to the original SRAM contents at the start of the test. Here the last march element is considered to "wrap around" to the first one - for example, consider programming a march sequence r-w\*-r\*-r\* - The ACSPAT shall be programmed as: 0b1001. Here ACSPAT[0] is 1 considering that the last element (corresponding to ACSPAT[3]) is an inverted read - and this "wraps around" to be the previous access of the first element, ACSPAT[0].

Here ACSPAT[0] corresponds to 'r' ; ACSPAT[1] corresponds to w\* and so on.

The address sequence is always linear, but can be incrementing or decrementing according to the setting of MCONTROL.DIR, and may be selected to change in bitline or wordline direction based on MCONTROL.RCADR.

## Memory Test Unit (MTU)

It is recommended that this test is run by setting MCONTROL.EN\_DESCR = 0 (i.e. over the system logical address space).

### Programming Sequences

Here a generic programming sequence on how to configure the NDT are described. It is assumed that the test is always run on the whole memory (i.e. the default value of RANGE is not changed).

Consider the sequence with 4 accesses (4N-NDT), {r, w\*, r\*, w}. For such a sequence, CONFIG0 and CONFIG1 have to be programmed to the following values:

CONFIG0 = 0x4005 (i.e. NUMACCS = 0x4; ACCSTYPE = 0b0101 -> Write-Read-Write-Read(1<sup>st</sup> access)).

CONFIG1 = 0x5008 (i.e. AG\_MOD = NDT; ACCSPAT = 0b1000 -> w (Previous Inverted Read) - r (Inverted Read) - w (Previous normal read) - r (normal read)).

Please note again that a 'w' access always writes the inverted value from the previous read access.

Test Programming Sequence:

1. Ensure that error detection is enabled, via the ALMSRCS and ECCS registers, and check for errors already present before the test (this is not required for the test, it is just a hint to software).  
Note that many steps here will trigger an UCE alarm & OPERR. In order to avoid any reaction from this expected error/alarm, the alarm reaction can be disabled before or ALMSRCS.OPENE can be set to 0 before starting the test.
2. Enter memory test mode (set the MTU\_MEMTEST.MEMxEN register).  
Note: This will trigger OPERR / UCE alarm. Additionally, for the security sensitive memories described in [Chapter 13.3.4.1](#) the test mode will be enabled only after the auto-initialization is complete. Therefore the software must wait for the MEMSTAT.AIUX bit to be cleared.
3. RANGE register assumed to have default reset value.
4. CONFIG0 = 0x4005
5. CONFIG1 = 0x5008  
Note: This will trigger OPERR / UCE alarm.
6. Set MCONTROL.DIR, RCADR and EN\_DESCR and start the test by writing a 1 to register bit MCONTROL.START. MCONTROL := 4009<sub>H</sub> (direction up, ROW first, EN\_DESCR = 0, START = 1). Note: This will trigger OPERR / UCE alarm if enabled.
7. Wait until MSTATUS.DONE is reset.
8. Clear MCONTROL.START  
MCONTROL := 4008<sub>H</sub> (direction up, ROW first, EN\_DESCR = 0, START = 0).
9. Wait for the end of the test - wait for MTU\_DONE interrupt to be triggered, or poll MSTATUS.DONE bit to be set, via MTU\_MEMDONE register.
10. Disable the memory test mode (clear the corresponding bit in the MTU\_MEMTEST register). For the security sensitive memories described in [Chapter 13.3.4.1](#) the test mode will be enabled only after the auto-initialization is complete. Therefore the software must wait for the MEMSTAT.AIUX bit to be cleared.
11. Check the result of the test  
- verify \*ERR bits in ECCD register
12. If the test failed check the error tracking registers ETRRx /ERRINFOx and the overflow bit (ECCD.EOF).
13. Clear the flags and the error tracking registers to enable further error tracking after the test.  
Clear the UCE alarm and OPERR flags set due to above operations, if they were enabled during the test.  
Otherwise, if UCE alarm reaction or ALMSRCS.OPENE were disabled before the test, re-enable them.

## Memory Test Unit (MTU)

To find failures during this test, software has to read the ECCD register for the CERR/UCERR and flags for any errors during the test. ETRR(0) will contain the first failed address, and the ERRINFO(0) register will contain the corresponding error type.

Consider a slightly more complicated sequence example, containing 8 access sequence in 3 programming steps:

$\{\{r, w^*\}, \{r^*, w, r, w^*\}, \{r^*, w\}\}$ . CONFIG0 and CONFIG1 for such a sequence shall be programmed as follows:

- $\{r, w^*\}$ : CONFIG0 = 0x2001 (i.e. NUMACCS = 2, ACCSTYPE = 0b01); CONFIG1 = 0x5000 (AG\_MOD = NDT, ACCSPAT = 0b00)
- $\{r^*, w, r, w^*\}$ : CONFIG0 = 0x4005 (i.e. NUMACCS = 4, ACCSTYPE = 0b0101); CONFIG1 = 0x5003 (AG\_MOD = NDT, ACCSPAT = 0b0110).
- $\{r^*, w\}$ : CONFIG0 = 0x2001 (i.e. NUMACCS = 2, ACCSTYPE = 0b01); CONFIG1 = 0x5003 (AG\_MOD = NDT, ACCSPAT = 0b10).

### Special Cases

For testing the complete CPU PMEM (i.e. PSPR + PCACHE area), the Non-Destructive test shall be run twice on the same memory, once with ECCS.TC\_TWR\_SEL = 0, and once with ECCS.TC\_TWR\_SEL = 1.

For EMEM SRAMs (except EMEM\_XTM), each read in the NDT is internally implemented by performing 2 reads. For example,  $\{r, w^*, r^*, w\}$  is actually performed as  $\{r, r, w^*, r^*, r^*, w\}$ . But this does not change the required programming of the registers (i.e. the CONFIG0 and CONFIG1 shall be programmed for  $\{r, w^*, r^*, w\}$  itself for example).

Special case of large DMEMs: In some devices of this family, certain CPUs have a large DMEM. Special handling is required in such a case. Please refer to the MTU chapter in the device specific appendix for a description.

### Usage of GANGs

The maximum current jump and total test time (for 4N-NDT) are specified in the datasheet of each device. In order to achieve these specified targets, the MBIST has to be run by grouping the SSHs into different gangs. Please refer to the appendix chapter of the device to find these Gangs.

Note: The total time for SRAM initialization (which is 1 access to each location, so a 1N operation) will be 1/4th of the time taken for the 4N test (this is specified in the datasheet of the device). However the maximum current is the same for 1N (e.g. SRAM intialization) or 4N (e.g. NDT) sequence.

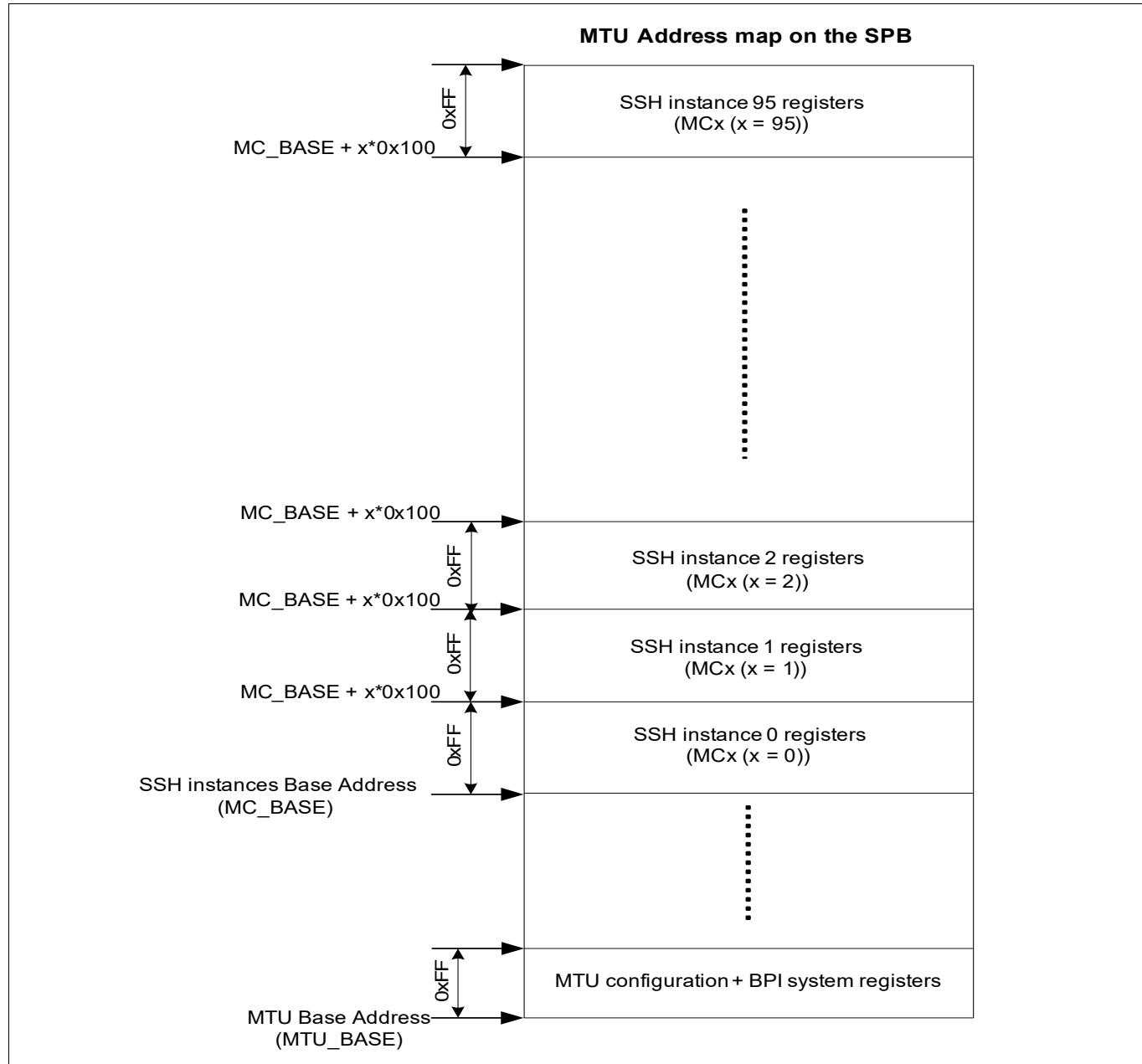
## 13.4 Registers

The overall address map of the various registers in the MTU is as shown in [Figure 154 “MTU Register Address Map” on Page 15](#). In addition to the standard system registers on the BPI, the registers in the MTU can be divided into two blocks:

- MTU Configuration registers - These registers provide the enable/disable functionality each individual memory controller.
- SRAM Support Hardware (SSH) Registers: These registers exist in each individual SSH in the device, and control the individual MBIST, ECC settings for the particular memory block.

*Note: Atomic bitwise operations are not supported on MTU/SSH registers.*

## **Memory Test Unit (MTU)**



**Figure 154 MTU Register Address Map**

**Table 476 Base addresses**

Name	Base Address	Description
MTU_BASE	0xF0060000	Base Address of the MTU.
MC_BASE	0xF0061000	Base Address of the SSH Instances.

**Memory Test Unit (MTU)****13.4.1 Registers Overview****Table 477 Register Overview - MTU (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
CLC	Clock Control Register	0000 <sub>H</sub>	U,SV	SV,E,P	Application Reset	<b>18</b>
ID	Identification Register	0008 <sub>H</sub>	U,SV	BE	Application Reset	<b>19</b>
MEMTESTi	Memory MBIST Enable Register i	0010 <sub>H</sub> +i*4	U,SV	SV,SE,P	Application Reset	<b>20</b>
MEMMAP	Memory Mapping Enable Register	001C <sub>H</sub>	U,SV	SV,SE,P	Application Reset	<b>21</b>
MEMSTATi	Memory Status Register i	0038 <sub>H</sub> +i*4	U,SV	BE	Application Reset	<b>22</b>
MEMDONEi	Memory Test Done Status Register i	0050 <sub>H</sub> +i*4	U,SV	BE	Application Reset	<b>23</b>
MEMFDAi	Memory Test FDA Status Register i	0060 <sub>H</sub> +i*4	U,SV	BE	Application Reset	<b>23</b>
ACCEN1	Access Enable Register 1	00F8 <sub>H</sub>	U,SV	BE	Application Reset	<b>20</b>
ACCENO	Access Enable Register 0	00FC <sub>H</sub>	U,SV	SV,SE	Application Reset	<b>19</b>
MCi_CONFIG0	Configuration Registers	1000 <sub>H</sub> +i*100 <sub>H</sub>	U,SV,16	U,SV,P,16	Application Reset	<b>24</b>
MCi_CONFIG1	Configuration Register 1	1002 <sub>H</sub> +i*100 <sub>H</sub>	U,SV,16	U,SV,P,16	Application Reset	<b>25</b>
MCi_MCONTROL	MBIST Control Register	1004 <sub>H</sub> +i*100 <sub>H</sub>	U,SV,16	SV,SE,P,16	Application Reset	<b>26</b>
MCi_MSTATUS	Status Register	1006 <sub>H</sub> +i*100 <sub>H</sub>	U,SV,16	BE	Application Reset	<b>29</b>
MCi_RANGE	Range Register, single address mode	1008 <sub>H</sub> +i*100 <sub>H</sub>	U,SV,16	U,SV,P,16	Application Reset	<b>30</b>
MCi_REVID	Revision ID Register	100C <sub>H</sub> +i*100 <sub>H</sub>	U,SV,16	BE	Application Reset	<b>31</b>
MCi_ECCS	ECC Safety Register	100E <sub>H</sub> +i*100 <sub>H</sub>	U,SV,16	SV,SE,P,16	Application Reset	<b>32</b>
MCi_ECCD	Memory ECC Detection Register	1010 <sub>H</sub> +i*100 <sub>H</sub>	U,SV,16	SV,P,16	See page <b>33</b>	<b>33</b>
MCi_ETRRx	Error Tracking Register x	1012 <sub>H</sub> +i*100 <sub>H</sub> +x*2	U,SV,16	BE	PowerOn Reset	<b>35</b>

**Memory Test Unit (MTU)****Table 477 Register Overview - MTU (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
MCi_RDBFLy	Read Data and Bit Flip Registry	1060 <sub>H</sub> +i* 100 <sub>H</sub> +y* 2	U,SV,16	U,SV,P,16	Application Reset	<b>36</b>
MCi_ALMSRCS	Alarm Sources Configuration Register	10EE <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,16	SV,SE,P,16	Application Reset	<b>37</b>
MCiFAULTSTS	SSH Safety Faults Status Register	10F0 <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,16	SV,SE,P,16	PowerOn Reset	<b>38</b>
MCiERRINFOx	Error Information Register x	10F2 <sub>H</sub> +i* 100 <sub>H</sub> +x* 2	U,SV,16	BE	PowerOn Reset	<b>40</b>

**Memory Test Unit (MTU)****13.4.2 Register Description**

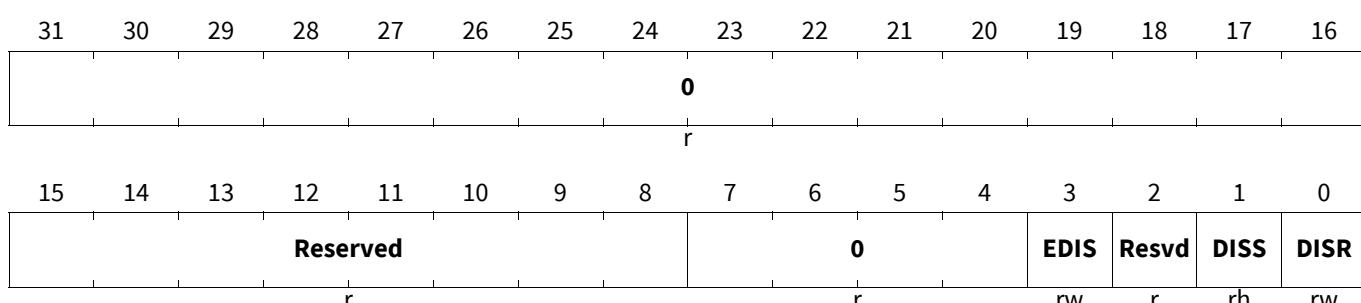
The following chapter describes the registers in the MTU. The MTU configuration registers described in [Page 20](#) control the enabling/disabling and autoinitialization functions of each memory controller. The memory controller registers described in [Page 24](#) are present for each memory controller in the device. They can be used to control and configure each memory controller separately.

**13.4.2.1 System Registers****Clock Control Register**

Whenever the clock to the MTU is disabled (either when CLC.DISR=1, or during sleep mode and CLC.EDIS=1,) then the alarms generated from the SSHs are not forwarded to the SMU. Therefore, if alarm notifications from the SSHs are required, the application should keep the MTU enabled. If the MTU is disabled, and an alarm occurs, the pending alarms are forwarded to the SMU when the MTU is re-enabled later.

**CLC**

**Clock Control Register** (0000<sub>H</sub>) Application Reset Value: 0000 0003<sub>H</sub>



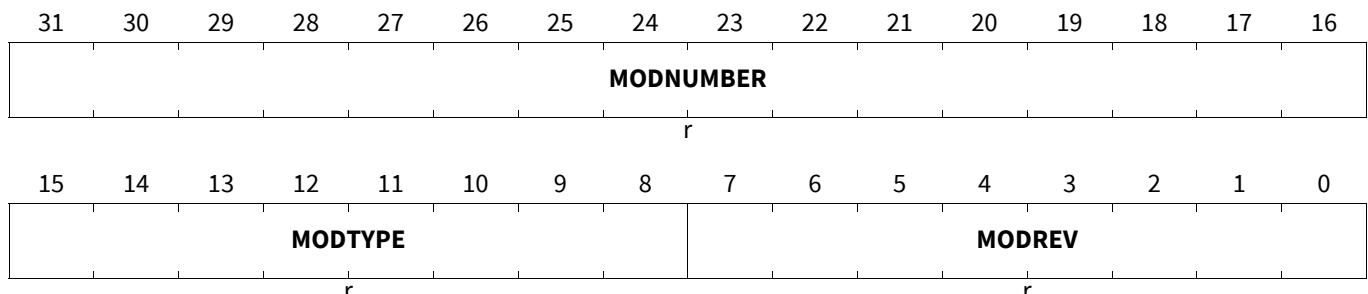
Field	Bits	Type	Description
<b>DISR</b>	0	rw	<b>Module Disable Request Bit</b> Used for enable/disable control of the module. 0 <sub>B</sub> Module disable is not requested 1 <sub>B</sub> Module disable is requested
<b>DISS</b>	1	rh	<b>Module Disable Status Bit</b> Bit indicates the current status of the module If the RMC field is implemented and if it is 0, DISS is set automatically. 0 <sub>B</sub> Module is enabled 1 <sub>B</sub> Module is disabled
<b>Resvd</b>	2	r	<b>Resvd</b> Read as 0. Must be written with 0 <sub>H</sub>
<b>EDIS</b>	3	rw	<b>Sleep Mode Enable Control</b> Used for module Sleep Mode control. 0 <sub>B</sub> Sleep Mode request is regarded. Module is enabled to go into Sleep Mode on a request. 1 <sub>B</sub> Sleep Mode request is disregarded: Sleep Mode cannot be entered on a request.
<b>Reserved</b>	15:8	r	<b>Reserved</b> Read as 0. Must be written with 0 <sub>H</sub>

## Memory Test Unit (MTU)

Field	Bits	Type	Description
0	7:4, 31:16	r	<b>0</b> Read as 0.

### Identification Register

**ID**  
**Identification Register** (0008<sub>H</sub>) Application Reset Value: 00B2 C003<sub>H</sub>



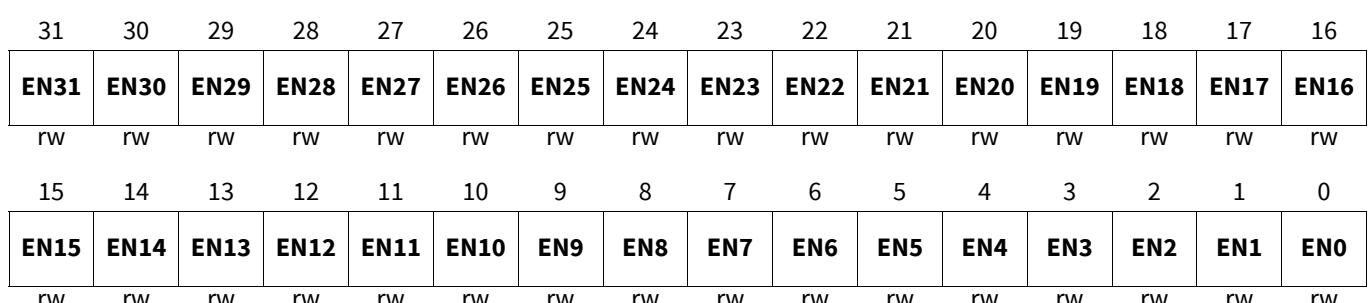
Field	Bits	Type	Description
<b>MODREV</b>	7:0	r	<b>Module Revision Number</b> This bit field indicates the revision number of the MTU module
<b>MODTYPE</b>	15:8	r	<b>Module Type</b> This bit field is C0 <sub>H</sub> . It defines a 32-bit module
<b>MODNUMBER</b>	31:16	r	<b>Module Number</b> This bit field defines the module identification number. The identification number for the AurixPlus Platform MTU module is 00B2 <sub>H</sub>

### Access Enable Register 0

The Access Enable Register 0 restricts write access to all MTU registers so that they may only be written by specified bus masters (eg CPUs). See the Bus chapter for the mapping of TAG ID to specific system masters and CPUs).

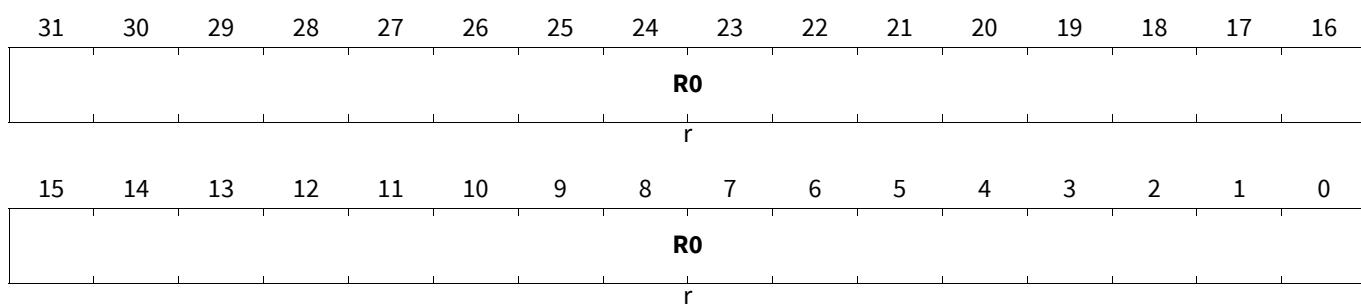
### ACCENO

**Access Enable Register 0** (00FC<sub>H</sub>) Application Reset Value: FFFF FFFF<sub>H</sub>



**Memory Test Unit (MTU)**

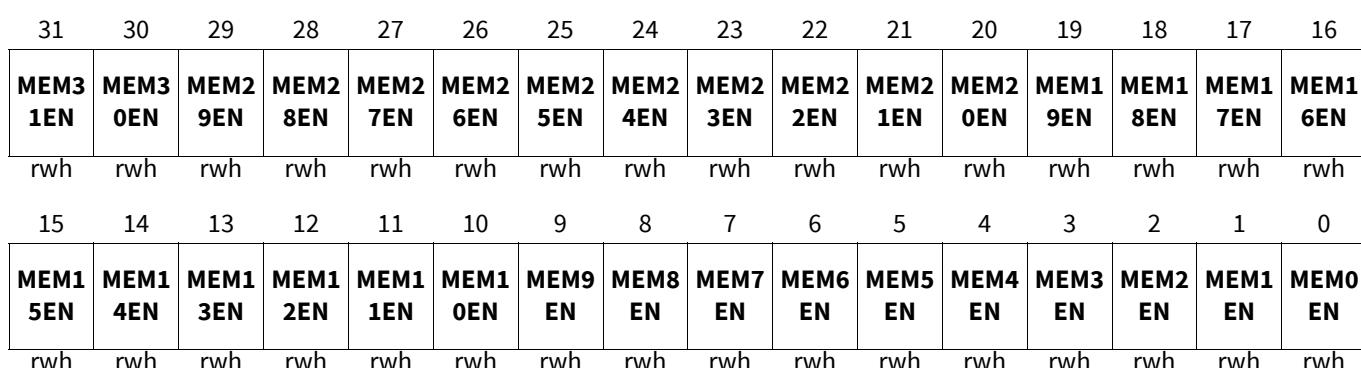
Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the MTU kernel addresses for transactions with the Master TAG ID n 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

**Access Enable Register 1****ACCEN1****Access Enable Register 1**(00F8<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>R0</b>	31:0	r	<b>Reserved - Res</b> Read as 0; should be written with 0.

**13.4.2.2 MTU Configuration Registers****Memory MBIST Enable Register i**

The memory test register MEMTEST holds CPU configurable select bits for the various SSH instances. See the product specific appendix for mapping of memory controller numbers.

**MEMTESTi (i=0-2)****Memory MBIST Enable Register i**(0010<sub>H</sub>+i\*4)Application Reset Value: 0000 0000<sub>H</sub>

## Memory Test Unit (MTU)

Field	Bits	Type	Description
MEMxEN (x=0-31)	x	rwh	<p><b>Memory x SSH instance Enable</b></p> <p>Security Notes:</p> <p>For bits which represent security-sensitive memories an automatic auto-initialization of the associated memory x is triggered on every attempt to toggle the MEMxEN . Only after this initialization has completed will the value read back from this register bit show the updated value. Register bit MEMSTAT.AIUX provides an indication that the automatic initialization of memory x is underway.</p> <p>See the Implementation Section for a list of memories which are security-sensitive</p> <p>0<sub>B</sub> Memory x SSH instance is disabled</p> <p>1<sub>B</sub> Memory x SSH instance is enabled</p>

## Memory Mapping Enable Register

The Memory Mapping Enable register MEMMAP has configurable control bits to select memory-mapped test mode. See the Integration Section for mapping of memory controller numbers.

### MEMMAP

#### Memory Mapping Enable Register

(001C<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>MEM3 1MAP</b>	<b>MEM3 0MAP</b>	<b>MEM2 9MAP</b>	<b>MEM2 8MAP</b>	<b>MEM2 7MAP</b>	<b>MEM2 6MAP</b>	<b>MEM2 5MAP</b>	<b>MEM2 4MAP</b>	<b>MEM2 3MAP</b>	<b>MEM2 2MAP</b>	<b>MEM2 1MAP</b>	<b>MEM2 0MAP</b>	<b>MEM1 9MAP</b>	<b>MEM1 8MAP</b>	<b>MEM1 7MAP</b>	<b>MEM1 6MAP</b>
rwh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>MEM1 5MAP</b>	<b>MEM1 4MAP</b>	<b>MEM1 3MAP</b>	<b>MEM1 2MAP</b>	<b>MEM1 1MAP</b>	<b>MEM1 0MAP</b>	<b>MEM9 MAP</b>	<b>MEM8 MAP</b>	<b>MEM7 MAP</b>	<b>MEM6 MAP</b>	<b>MEM5 MAP</b>	<b>MEM4 MAP</b>	<b>MEM3 MAP</b>	<b>MEM2 MAP</b>	<b>MEM1 MAP</b>	<b>MEM0 MAP</b>
rwh															

**Memory Test Unit (MTU)**

Field	Bits	Type	Description
<b>MEMxMAP (x=0-31)</b>	x	rwh	<p><b>MEMx Mapping Enable</b>  Note that only CPU Cache Memory Mapping bits are implemented (See Implementation Section for details of used bits in this product)</p> <p><b>Security Notes:</b>  Caches are considered security-sensitive memories and an automatic auto-initialization (or partial erase) of the associated memory x is triggered on every attempt to toggle the MEMxMAP bit <sup>1)</sup>. Only after this initialization has completed will the value read back from this register bit show the updated value. Register bit MEMSTAT.AIUX provides an indication that the automatic initialization of memory x is underway.</p> <p>0<sub>B</sub> Memory x functional  1<sub>B</sub> Memory x memory-mapped (e.g. for test)</p>

1)

**Memory Status Register i**

The memory status register MEMSTAT shows whether each SSH instance is currently executing an automatic initialization sequence.

**MEMSTATi (i=0-2)****Memory Status Register i**(0038<sub>H</sub>+i\*4)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>MEM3 1AIU</b>	<b>MEM3 0AIU</b>	<b>MEM2 9AIU</b>	<b>MEM2 8AIU</b>	<b>MEM2 7AIU</b>	<b>MEM2 6AIU</b>	<b>MEM2 5AIU</b>	<b>MEM2 4AIU</b>	<b>MEM2 3AIU</b>	<b>MEM2 2AIU</b>	<b>MEM2 1AIU</b>	<b>MEM2 0AIU</b>	<b>MEM1 9AIU</b>	<b>MEM1 8AIU</b>	<b>MEM1 7AIU</b>	<b>MEM1 6AIU</b>
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>MEM1 5AIU</b>	<b>MEM1 4AIU</b>	<b>MEM1 3AIU</b>	<b>MEM1 2AIU</b>	<b>MEM1 1AIU</b>	<b>MEM1 0AIU</b>	<b>MEM9 AIU</b>	<b>MEM8 AIU</b>	<b>MEM7 AIU</b>	<b>MEM6 AIU</b>	<b>MEM5 AIU</b>	<b>MEM4 AIU</b>	<b>MEM3 AIU</b>	<b>MEM2 AIU</b>	<b>MEM1 AIU</b>	<b>MEM0 AIU</b>
rh															

Field	Bits	Type	Description
<b>MEMxAIU (x=0-31)</b>	x	rh	<p><b>Memory x MBIST AutoInitialize Underway</b>  This bit indicates whether an automatic data initialization (or partial erase) of Memory x has been triggered by a change of state of MEMTEST.MEMxEN or MEMxMAP but that the initialization sequence has not yet completed.</p> <p>0<sub>B</sub> Memory x SSH instance not running autoinitialize  1<sub>B</sub> Memory x SSH instance running autoinitialize</p>

**Memory Test Unit (MTU)****Memory Test Done Status Register i**

Each bit in one of the memory test done status registers MEMDONE<sub>i</sub> reflects the status of the MSTATUS.DONE bit in the corresponding SSH. See the implementation section for the implemented register bits.

**MEMDONE<sub>i</sub> (i=0-2)****Memory Test Done Status Register i**(0050<sub>H</sub>+i\*4)Application Reset Value: FFFF FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEM3 1DON E	MEM3 0DON E	MEM2 9DON E	MEM2 8DON E	MEM2 7DON E	MEM2 6DON E	MEM2 5DON E	MEM2 4DON E	MEM2 3DON E	MEM2 2DON E	MEM2 1DON E	MEM2 0DON E	MEM1 9DON E	MEM1 8DON E	MEM1 7DON E	MEM1 6DON E
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEM1 5DON E	MEM1 4DON E	MEM1 3DON E	MEM1 2DON E	MEM1 1DON E	MEM1 0DON E	MEM9 DONE	MEM8 DONE	MEM7 DONE	MEM6 DONE	MEM5 DONE	MEM4 DONE	MEM3 DONE	MEM2 DONE	MEM1 DONE	MEM0 DONE
rh															

Field	Bits	Type	Description
MEMzDONE (z=0-31)	z	rh	<b>Memory SSH MSTATUS.DONE</b> 0 <sub>B</sub> Memory SSH MSTATUS.DONE = 0 1 <sub>B</sub> Memory SSH MSTATUS.DONE = 1

**Memory Test FDA Status Register i**

Each bit in one of the memory test done status registers MEMFDA<sub>i</sub> reflects the status of the MSTATUS.FDA bit in the corresponding SSH. See the implementation section for the implemented register bits.

**MEMFDA<sub>i</sub> (i=0-2)****Memory Test FDA Status Register i**(0060<sub>H</sub>+i\*4)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEM3 1FDA	MEM3 0FDA	MEM2 9FDA	MEM2 8FDA	MEM2 7FDA	MEM2 6FDA	MEM2 5FDA	MEM2 4FDA	MEM2 3FDA	MEM2 2FDA	MEM2 1FDA	MEM2 0FDA	MEM1 9FDA	MEM1 8FDA	MEM1 7FDA	MEM1 6FDA
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEM1 5FDA	MEM1 4FDA	MEM1 3FDA	MEM1 2FDA	MEM1 1FDA	MEM1 0FDA	MEM9 FDA	MEM8 FDA	MEM7 FDA	MEM6 FDA	MEM5 FDA	MEM4 FDA	MEM3 FDA	MEM2 FDA	MEM1 FDA	MEM0 FDA
rh															

## Memory Test Unit (MTU)

Field	Bits	Type	Description
<b>MEMzFDA (z=0-31)</b>	z	rh	<b>Memory SSH MSTATUS.FDA</b> 0 <sub>B</sub> Memory SSH MSTATUS.FDA = 0 1 <sub>B</sub> Memory SSH MSTATUS.FDA = 1

### 13.4.2.3 SRAM Support Hardware (SSH) Registers

There is one set of registers for each SSH instance corresponding to each SSH instance (please refer to the appendix chapter for the list of SSH instances). These registers are described below.

Some register field sizes or content depend upon the physical sizes of the memories. The default settings enable fill/test of the entire physical RAM and the register descriptions show the maximum bitfield sizes.

#### Configuration Registers

The bits in these registers can be used to control and program any march and hammer sequences. All bits concerning these test are concentrated here. All bits do not change during a test run. Setting MCONTROL.START will start the tests defined here. MSTATUS.DONE is reset at the beginning of a test and set after completion once MCONTROL.START is cleared by software. If no legal operation was defined in CONFIG1.AG\_MOD nothing will be done but the handshake of MCONTROL.START and MSTATUS.DONE is carried out.

The reset values of the CONFIG0/1 and MCONTROL registers will perform a {↑(w0,r0)} operation (direction up, write 0 to all cells and check for 0) which initializes the whole memory with 0 and checks the contents if MCONTROL.START is set. This is the start sequence of many tests.

#### MCi\_CONFIG0 (i=0-95)

##### Configuration Registers

(1000<sub>H</sub>+i\*100<sub>H</sub>)

Application Reset Value: 2002<sub>H</sub>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NUMACCS				R8				ACCSTYPE							
rw				r				rw							

Field	Bits	Type	Description
<b>ACCSTYPE</b>	7:0	rw	<b>Access type</b> This field specifies the type of access which is being performed to each single address in the current marching element. ACCSTYPE[n] specifies the n-th access of the marching element. 0 <sub>b</sub> write access 1 <sub>b</sub> read access
<b>R8</b>	11:8	r	<b>Reserved - Res</b> Reads return 0. Writes have no effect.

## Memory Test Unit (MTU)

Field	Bits	Type	Description
NUMACCS	15:12	rw	<p><b>Number of accesses per address</b></p> <p>This field specifies the total number of accesses which are being performed to each single address in the current marching element.</p> <p>Allowed values: 0-8 (Due to size limitation of CONFIG0.ACCTYPE and CONFIG1.ACSPAT fields).</p> <p>If NUMACCS=0 will not access a memory.</p> <p>If NUMACCS &gt; 8, 8 accesses will be performed.</p>

## Configuration Register 1

## MCi\_CONFIG1 (i=0-95)

## Configuration Register 1

(1002<sub>H</sub>+i\*100<sub>H</sub>)Application Reset Value: 0000<sub>H</sub>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AG_MOD				SELFASB				ACCSPAT							
rw				rw				rw							

Field	Bits	Type	Description
ACCSPAT	7:0	rw	<p><b>Access pattern</b></p> <p>When AG_MOD is selected for any test other than the Non-Destructive test, this field specifies directly the bit pattern (i.e. '0' or '1') which is being used for an access to each single address in the current marching element.</p> <p>ACCSPAT[n] specifies the n-th access of the marching element. These patterns are toggled according to MCONTROL.BITTOG and MCONTROL.ROWTOG.</p> <p>When AG_MOD selects the Non-Destructive test: For corresponding ACCSTYPE as READ or WRITE access:</p> <p>Program 0 when the previous read access was with normal data; and 1 when the previous read was with inverted data.</p> <p>Note: When considering the previous read access, consider that the last access is a previous access to the first, as a "wrap around".</p> <p>Please refer to section on Non-Destructive test for more details on how to program these bits.</p>
SELFASB	11:8	rw	<p><b>Select Fast Bit</b></p> <p>This field defines during a <math>2^i</math> test the address bit position that has the Hamming distance of 1, i. e. changes fastest. Bit 0 of either column or row address is swapped with the indicated bit of either column or row according to MCONTROL.RCADR.</p> <p>MCONTROL.RCADR=0 -&gt; column MCONTROL.RCADR=1 -&gt; row</p> <p><math>0_H</math> normal addressing sequence, bit 0 in its normal position. <b>others</b>, bit 0 swapped with the indicated position.</p>

## Memory Test Unit (MTU)

Field	Bits	Type	Description
<b>AG_MOD</b>	15:12	rw	<p><b>Address Generator Mode</b></p> <p>These bits enable the special hardware for performing the more complex addressing schemes.</p> <p>In case RANGE.RAEN (range enable) is set to 0 (single access) linear address mode has to be selected and NUMACCS set to 1.</p> <ul style="list-style-type: none"> <li><math>0_H</math> run the test with linear address generation</li> <li><math>1_H</math> run the right half select test</li> <li><math>2_H</math> run the test with GALPAT9 algorithm</li> <li><math>3_H</math> run the left half select test</li> <li><math>4_H</math> run the test with the GALPAT5 algorithm</li> <li><math>5_H</math> run the non-destructive test. The march elements, direction and backgrounds are defined by other settings in CONFIG0/1 and MCONTROL. For this test, the SRAM has to be pre-initialized with valid content (i.e. with ECC correct data). Unlike a normal MBIST march test, this test uses the ECC itself to find errors in the data. The result of the test is not reflected via MSTATUS.FAIL - instead, the detected ECC errors are tracked in the ETRR &amp; ERRINFO registers, and additionally ECCD:*ERR bits if the alarm notifications are enabled. registers</li> <li><math>8_H</math> run the write mask test</li> <li><math>A_H</math> run the test with <math>2^i</math> address generation</li> <li><b>others</b>, Nothing is done but the handshake of START and DONE is carried out.</li> </ul>

### MBIST Control Register

The bits in these registers can be used to control and program any march and hammer sequences. All bits concerning these test are concentrated here. All bits do not change during a test run. Setting MCONTROL.START will start the tests defined here. MSTATUS.DONE is reset at the beginning of a test and set after completion once MCONTROL.START is cleared by software. If no legal operation was defined in CONFIG1.AG\_MOD nothing will be done but the handshake of MCONTROL.START and MSTATUS.DONE is carried out.

The reset values of the CONFIG0/1 and MCONTROL registers will perform a  $\{\uparrow(w0,r0)\}$  operation (direction up, write 0 to all cells and check for 0) which initializes the whole memory with 0 and checks the contents if MCONTROL.START is set. This is the start sequence of many tests.

#### MCi\_MCONTROL (i=0-95)

#### MBIST Control Register

( $1004_H + i * 100_H$ )

Application Reset Value:  $4008_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>SRAM_CLR</b>	<b>R14</b>	<b>R13</b>	<b>R12</b>	<b>R11</b>	<b>EN_DE SCR</b>	<b>FAILD MP</b>	<b>R8</b>	<b>BITTO G</b>	<b>ROWT OG</b>	<b>RCADR</b>	<b>DINIT</b>	<b>DIR</b>	<b>ESTF</b>	<b>RESUME</b>	<b>START</b>

**Memory Test Unit (MTU)**

Field	Bits	Type	Description
<b>START</b>	0	rw	<p><b>START</b></p> <p>If this bit is written to '1' by software the memory test will start. If it is reset by software, and the test has finished, MSTATUS.DONE will be set to 1.</p> <p>If MCONTROL.FAILDMP is set, a fail will stop the current execution. RESUME will continue a suspended test.</p> <p>0<sub>B</sub> No test started, finished or waiting for test end 1<sub>B</sub> Start memory test</p>
<b>RESUME</b>	1	rwh	<p><b>Resume failed test</b></p> <p>This bit allows a test with fail that got suspended to be resumed after the dump of the fail bit map. A restart is possible only if MSTATUS.FDA was reset by hardware. It will be reset by hardware once the test is resumed.</p> <p>0<sub>B</sub> Do not resume 1<sub>B</sub> Resume suspended MBIST run</p>
<b>ESTF</b>	2	rw	<p><b>Enable Sticky Fail Bit</b></p> <p>This bit enables the sticky fail bit MSTATUS.SFAIL. If set any fails will be collected in MSTATUS.SFAIL. Resetting this bit to 0 will also reset MSTATUS.SFAIL.</p> <p>0<sub>B</sub> Do not collect fail events 1<sub>B</sub> Collect fail events</p>
<b>DIR</b>	3	rw	<p><b>Direction Select</b></p> <p>This field specifies the direction of a memory test operation.</p> <p>0<sub>B</sub> DOWN: Address direction is highest to lowest. 1<sub>B</sub> UP: Address direction is lowest to highest.</p>
<b>DINIT</b>	4	rw	<p><b>Data Initialization Enable</b></p> <p>This bit enables a write of the RDBFL data to all locations defined by the range register. RDBFL can contain data that will produce an ECC error. Execution is started with MCONTROL.START. For this predefined action any information contained in CONFIG0/1 registers and the bits BITTOG, ROWTOG and DIR are ignored.</p> <p>0<sub>B</sub> Disabled 1<sub>B</sub> Enabled</p>
<b>RCADR</b>	5	rw	<p><b>Fast Row / Fast Column Addressing Scheme Select</b></p> <p>This bit selects between fast row and fast column addressing. "Fast Row" moves along the word-lines first and then in bit-line direction, "Fast Column" along the bit-lines first.</p> <p>0<sub>B</sub> Fast row 1<sub>B</sub> Fast column</p>
<b>ROWTOG</b>	6	rw	<p><b>Row toggling</b></p> <p>This field specifies whether to toggle the used bit pattern (non inverted/inverted) with each physical memory row. This is required when writing a checkerboard pattern or a row stripe pattern.</p> <p>0<sub>B</sub> Do not toggle 1<sub>B</sub> Do toggle with each row</p>

## Memory Test Unit (MTU)

Field	Bits	Type	Description
<b>BITTOG</b>	7	rw	<p><b>Bit toggling</b></p> <p>This field specifies whether to toggle the used bit pattern (non inverted/inverted) with each physical memory column. This is required when writing a checkerboard pattern or a column stripe pattern.</p> <p>0<sub>B</sub> Do not toggle 1<sub>B</sub> Do toggle with each column</p>
<b>R8</b>	8	rw	<p><b>Reserved</b></p> <p>This bit shall always be written with 0.</p>
<b>FAILDMP</b>	9	rw	<p><b>Fail bitmap dump</b></p> <p>This field enables a dump of the failing address and a fail bit map after a fault has been detected. The memory test is suspended afterwards and resumed by MCONTROL.RESUME. MSTATUS.FDA shows that a fail dump is available.</p> <p>This functionality can be used only if bit MCONTROL.LDRED = 1. In case a fail dump is available, RDBFL will contain the fail bit map and ETRR the failing address.</p> <p>0<sub>B</sub> Do not dump 1<sub>B</sub> Dump each fault</p>
<b>EN_DESCR</b>	10	rw	<p><b>Enable Descrambling</b></p> <p>This bit has an effect only when the SSH itself is enabled. If this bit is set, the internal address de-scrambler in the SSH will be enabled. The reset value is 0, hence the de-scrambler is not enabled by default .</p> <p>0<sub>B</sub> Descrambler is not enabled in the address generation path within the SSH 1<sub>B</sub> Descrambler is enabled in the address generation path within the SSH</p>
<b>R11</b>	11	r	<p><b>Reserved</b></p> <p>Reads return 0</p>
<b>R12</b>	12	rw	<p><b>Reserved</b></p> <p>This bit shall always be written with 0.</p>
<b>R13</b>	13	rw	<p><b>Reserved</b></p> <p>This bit shall always be written with 0.</p>
<b>R14</b>	14	rw	<p><b>Reserved</b></p> <p>This bit shall always be written with 1.</p>
<b>SRAM_CLR</b>	15	rw	<p><b>Clear the SRAM</b></p> <p>This bit initializes the complete SRAM with ECC correct "All-0" data. Execution is started with MCONTROL.START. For this predefined action any information contained in CONFIG0/1, RANGE registers and the bits BITTOG, ROWTOG and DIR are ignored. This bit shall not be set together with other initialization or test configurations. After the SRAM clearing is complete, the software has to reset this bit back to '0' before disabling the SSH.</p> <p>0<sub>B</sub> Do not clear the entire SRAM. 1<sub>B</sub> Clear the entire SRAM. The SRAM is fully filled with zeroes, and is also ECC correct.</p>

## Memory Test Unit (MTU)

### Status Register

The bits in the status register show the status of the currently running and last test respectively.

#### MCI\_MSTATUS (i=0-95)

##### Status Register

( $1006_H + i * 100_H$ )

Application Reset Value:  $0001_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R4										rh	Res4	SFAIL	FDA	FAIL	DONE

Field	Bits	Type	Description
<b>DONE</b>	0	rh	<b>DONE</b> This bit is reset at the start of a test and set when a test is completed and MCONTROL.START was reset by software. It is not set when a test is interrupted for fail dump.
<b>FAIL</b>	1	rh	<b>FAIL</b> This bit will be reset when a test is being started. It will be set to '1' by hardware under the following conditions: 0 <sub>B</sub> no error occurred 1 <sub>B</sub> detailed description see above
<b>FDA</b>	2	rh	<b>Fail Dump Available</b> This bit shows that a fail has occurred if MCONTROL.FAILDMP is set. The test is suspended and fail dump information is available. The fail bit map is in RDBFL and the associated address is in ETRR(0). As long as no fail has occurred RDBFL contains the last read information and ETRR has no valid data. This bit will be set by hardware. It will be reset when MSTATUS was read with MSTATUS.FDA = 1 and the dump information was read from ETRR and RDBFL. Only the last read from the last word of RDBFL is checked by the hardware and taken as an indication for a complete read. A suspended test will be resumed by MCONTROL.RESUME if FDA was reset. This forms some sort of handshake to insure that a suspended test can only be resumed (by a broadcasted) MCONTROL.RESUME if the last fail information was actually collected. 0 <sub>B</sub> No fail dump data available. A suspended MBIST run can be resumed. 1 <sub>B</sub> Fail dump data is available and waiting for read.
<b>SFAIL</b>	3	rh	<b>Sticky Fail Bit</b> This bit is set to 1 together with MSTATUS.FAIL provided MCONTROL.ESTF is set. In contrast to FAIL it will not be reset when a new test is started. Therefore it will collect fail information over more than one MBIST run. It will be reset when MCONTROL.ESTF is reset, or MBIST mode is switched off. 0 <sub>B</sub> No fail collected. 1 <sub>B</sub> A fail occurred during one of the test runs since MCONTROL.ESTF was set to 1.

## Memory Test Unit (MTU)

Field	Bits	Type	Description
Res4	4	rh	<b>Reserved - Res</b> Shall be written with zero.
R4	15:5	r	<b>Reserved - Res</b> Reads return 0

### Range Register, single address mode

The Range Register can be used to run a test only on a dedicated part of the RAM.

The range can be set in 64 word increments.

The range register can also be used to write to one specific address or read from it. In this case the range is disabled and the remaining part of the register is used as the address field. The addresses generated via the RANGE register (single or range) can be physical or logical depending on the MCONTROL.EN\_DESCR bit.

#### MCi\_RANGE (i=0-95)

		(1008 <sub>H</sub> +i*100 <sub>H</sub> )										Application Reset Value: XXXX <sub>H</sub>						
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
RAEN	INJERR	ADDR																
rw	rw									rw								

Field	Bits	Type	Description
ADDR	13:0	rw	<b>Address</b> When RAEN = 0, This field specifies the address of a single memory location. Reads and writes to this location are possible. When RAEN=1, this field is interpreted as 2 different fields. ADDR[13:7] is interpreted as Upper Range Limit. ADDR[6:0] is interpreted as Lower Range Limit. For smaller SRAMs which require lesser number of address bits, the MSB bits are reserved. Writes to these bits are ignored, and reads return '0'.
INJERR	14	rw	<b>Inject Error</b> Enables Error-Injection during march tests. This is supported only for linear march tests. 0 <sub>B</sub> Do not mask any writes during march tests. RANGE.ADDR and RAEN used normally. 1 <sub>B</sub> Use RANGE.ADDR as a pointer to a physical SRAM address to which write accesses during a march test will not be executed. This bit helps in error injection during a march test over the whole SRAM. This bit has an effect only when RAEN is also set. With INJERR and RAEN = '1', the test is by default run over the entire SRAM.

## Memory Test Unit (MTU)

Field	Bits	Type	Description
RAEN	15	rw	<p><b>Range Enable</b></p> <p>0 Disabled, single address mode. In this case a single word can be addressed for read or write. Config registers have to be set as follows      CONFIG.NUMACCS:= “0001” (single access)      CONFIG.AG_MOD := “0000” (linear)      MCONTROL.DIR :=1 (up)</p> <p>For read just the value in this location will be delivered. No check against expected values is made; i.e. MSTATUS.FAIL will not be set.</p> <p>1 Enabled. ADDR[13:7] is interpreted as Upper Range Limit. ADDR[6:0] is interpreted as Lower Range Limit.</p>

### Setting Address Ranges

If the RAEN field is set to ‘1’ then range mode is enabled. In this case, the ADDR field of the RANGE register is interpreted as two separate fields:

ADDR[13:7] is interpreted as the Upper Range Limit.

This field specifies the upper logical block address limit in 64 word increments.Upper end of the address range is UPLIMIT & 111111B

ADDR[6:0] is interpreted as the Lower Range Limit

This field specifies the lowerlogical block address limit in 64 word increments.Lower end of the address range is LOLIMIT & 000000B.

Note that the default reset value of the ADDR field will be set to the maximum range of the physical memory. The default behaviour is therefore that an initialization or test will operate over the whole memory.

Also note that for smaller memories which require less than 13 bits of addressing, the relevant MSB bits of the address field are reserved.

### Usage of MCi\_RANGE register

Usage of MCi\_RANGE register in order to perform initialization or MBIST over any partial SRAM address ranges is not recommended. The recommendation is to always perform initialization and MBIST over the complete address range of any SRAM. Therefore the user is recommended not to change the reset value of this register, and always use this register with its reset value while performing any tests or initialization.

### Revision ID Register

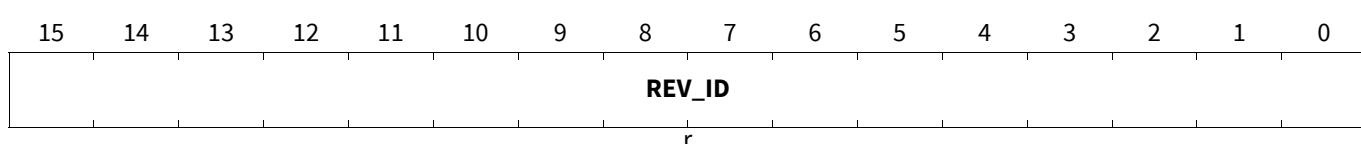
The revision ID register contains a hard coded read only constant which describes the current status of the MBIST/ECC IP.

#### MCi\_REVID (i=0-95)

#### Revision ID Register

(100C<sub>H</sub>+i\*100<sub>H</sub>)

Application Reset Value: 0610<sub>H</sub>



## Memory Test Unit (MTU)

Field	Bits	Type	Description
<b>REV_ID</b>	15:0	r	<b>Revision Identifier</b> This field defines the currently implemented release, version and functionality of the used MBIST/ECC controller to track the MBIST/ECC version for easier handling at the tester.

### ECC Safety Register

This register controls the various error detection and notification modes. This register can be accessed even if the corresponding SSH is not enabled using the MEMTEST.MEMx\_EN bit.

Writing to this register is only permitted when safety endinit is cleared.

The hardware features that implement fault tolerance mechanisms for the SRAMs (e.g. single bit correction) shall be enabled per default after any reset. This is ensured by the reset value of the ECC safety register where ECCS.ECE = 1 after a reset.

#### MCI\_ECCS (i=0-95)

<b>ECC Safety Register</b>												<b>(100E<sub>H</sub>+i*100<sub>H</sub>)</b>				<b>Application Reset Value: 001F<sub>H</sub></b>			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
R12			SFFD	TC_T WR_S EL	ECCMAP		R7	SFLE	BFLE	TRE	ECE	MENE	UCEN E	CENE					
r			rwh	rw		rw	r	rw	rw	rw	rw	rw	rw	rw	rw				

Field	Bits	Type	Description
<b>CENE</b>	0	rw	<b>ECC Correction Event Alarm Notification Enable</b> This bit enables the forwarding of the CE alarm from the SSH to the SMU. 0 <sub>B</sub> Do not forward CE alarm to SMU. 1 <sub>B</sub> Forward the CE alarm to SMU.
<b>UCENE</b>	1	rw	<b>Uncorrectable Error Affecting SRAM / SSH Operation: Alarm Notification Enable - UENE</b> This bit enables the forwarding of the UCE alarm from the SSH to the SMU. Please refer to the section on safety for more details. 0 <sub>B</sub> Do not forward the UCE alarm to the SMU. 1 <sub>B</sub> Forward the UE alarm to the SMU
<b>MENE</b>	2	rw	<b>Miscellaneous Alarm Notification Enable: MENE</b> This bit enables the forwarding of the ME alarm from the SSH to the SMU. Please refer to the section on safety for more details. 0 <sub>B</sub> Do not forward the ME alarm to the SMU. 1 <sub>B</sub> Forward the ME alarm to the SMU.
<b>ECE</b>	3	rw	<b>Error Correction Enable</b> This enables the single bit error correction by the ECC. If this bit is 1, single bit errors are flagged via the CE alarm. If this bit is 0, single bit errors are flagged via the UE alarm. 0 <sub>B</sub> Do not correct correctable errors. 1 <sub>B</sub> Correct correctable errors

**Memory Test Unit (MTU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>TRE</b>	4	rw	<p><b>Tracking Enable</b>            All errors will be tracked, if the associated notification enable bit is set.            This bit is enabled by default.</p> <p><math>0_B</math> Do not track address of detected error.  <math>1_B</math> Track address of detected error.</p>
<b>BFLE</b>	5	rw	<p><b>Bit Flip Enable</b>  <math>0_B</math> Normal operation.  <math>1_B</math> Test mode only. Flips data and check bits according to RDBFL.</p>
<b>SFLE</b>	6	rw	<p><b>Signature Bit Flip Enables</b>            If address error detection is enabled (ALMSRCS.ADDRE = 1) and If this bit is set and the SRAM is read, an address error is notified, and tracked in the ETRR &amp; ERRINFO registers, as well as an alarm is generated, if enabled.            Note that for SRAMs with Address-ECC (refer the Appendix chapter for the list), this bit is ignored, and no error will be generated.</p> <p><math>0_B</math> Do not force address error injection.  <math>1_B</math> Forces address error injection by flipping bit[0] of the address to the address error detection logic, but not to the SRAM. This results in an address error to be generated.</p>
<b>R7</b>	7	r	<p><b>Reserved - Res</b>            Write 0; Reads return 0</p>
<b>ECCMAP</b>	9:8	rw	<p><b>ECC Bit Mapping Mode</b>            ECCMAP sets three different test modes to allow access to data or ECC bits separately and independently.</p> <p><math>00_B</math> Normal operation  <math>01_B</math> Test mode. Only data bits mapped. All ECC functionality disabled.  <math>10_B</math> Test mode. ECC check bits mapped to lower data bit positions.            Other bits read as zero. All ECC functionality disabled. Data bits are not affected by write operations.  <math>11_B</math> Do not use this setting.</p>
<b>TC_TWR_SEL</b>	10	rw	<p><b>TriCore Tower Select</b>            For TriCore PMEM only. This bit selects a cache way to run the non-destructive inversion test on. This bit represents the Tower number.</p>
<b>SFFD</b>	11	rwh	<p><b>Safety Flip-Flop Diagnostics</b>            Safety Flip-Flop Diagnostics bit. Setting this bit triggers a Safety Flip-Flop self test. The result of the test (i.e. any error status in the safety FFs) - can be obtained from the OPERR or MISCERR bits in the FAULTSTS register.</p> <p><math>0_B</math> Do not trigger an SFF self test.  <math>1_B</math> Trigger an SFF self test. Bit is cleared automatically by the hardware when the test is complete.</p>
<b>R12</b>	15:12	r	<p><b>Reserved - Res</b>            Reads return 0</p>

**Memory ECC Detection Register**

The ECC detection register contains information on the errors detected and the tracking register clear.

**Memory Test Unit (MTU)****MCi\_ECCD (i=0-95)****Memory ECC Detection Register****(1010<sub>H</sub>+i\*100<sub>H</sub>)****Reset Value: Table 478**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EOV</b>	<b>PERMERR</b>				<b>VAL</b>				<b>TRC</b>	<b>MERR</b>	<b>UCER R</b>	<b>CERR</b>	<b>SERR</b>		

rh                    rw                    rh                    w                    rwh                    rwh                    rwh                    rwh

Field	Bits	Type	Description
<b>SERR</b>	0	rwh	<b>Error Detected</b> Write of '0' clears the sticky status. Write of '1' has no effect. In the case of a write of '0' simultaneously with an error detection, the setting of the bit by hardware will take priority. This bit is reset with an Application Reset. Read as: 0 <sub>B</sub> No error detected. 1 <sub>B</sub> An error was detected and alarm forwarded: CERR, UCERR or MERR.
<b>CERR</b>	1	rwh	<b>CE alarm occurred</b> Write of '0' clears the bit, and enables further alarms to be forwarded to SMU. Write of '1' has no effect. When the bit is set, software can perform additional diagnostics from the information in the ETRR/ERRINFO registers. Please refer to the safety section for more details. This bit is reset with an Application Reset. Read as: 0 <sub>B</sub> No CE alarm event occurred. 1 <sub>B</sub> CE alarm event occurred.
<b>UCERR</b>	2	rwh	<b>Uncorrectable Error Alarm Occured</b> Write of '0' clears the bit, and enables further alarms to be forwarded to SMU. When the bit is set, software can perform additional diagnostics from the information in the ETRR/ERRINFO registers. Please refer to the safety section for more details. Write of '1' has no effect. This bit is cleared on an application reset. Read as: 0 <sub>B</sub> No UCE alarm event occurred. 1 <sub>B</sub> UCE alarm event occurred
<b>MERR</b>	3	rwh	<b>Miscellaneous Error Alarm Occured</b> Write of '0' clears the bit, and enables further alarms to be forwarded to SMU. When the bit is set, software can perform additional diagnostics from the information in the ETRR/ERRINFO and ALMSRCS registers. Please refer to the safety section for more details. Write of '1' has no effect. This bit is reset with an application reset. Read as: 0 <sub>B</sub> No ME Alarm Event occurred. 1 <sub>B</sub> ME Alarm Event occurred.

## Memory Test Unit (MTU)

Field	Bits	Type	Description
TRC	4	w	<p><b>Tracking Clear</b></p> <p>Writing this bit with '1' clears the EOV, VAL bits plus the ETRR and ERRINFO registers, depending on the PERMERR settings.</p> <p>This bit will always read 0.</p> <p><math>0_B</math> No effect.</p> <p><math>1_B</math> Clear the ETRR, ERRINFO and ECCD.VAL &amp; EOV bits. If a PERMERR bit is set, then the corresponding entries are not cleared. Note: If PERMERR and TRC are written at the same time, the clearing due to TRC takes place with the previous PERMERR settings, and the new settings take effect only after.</p>
VAL	9:5	rh	<p><b>Valid Bits</b></p> <p>Every tracking register (ETRRx) has a valid bit associated. Reset by ECCD.TRC. 5 error tracking registers are available and 5 valid bits. These bits are preserved until a PORST.</p>
PERMERR	14:10	rw	<p><b>Permanent Error in ETRR Entry</b></p> <p>Denotes an ETRR entry that shall not be cleared by setting the TRC or moved up when a new error occurs. With this bit set, the corresponding ETRR+ERRINFO entry remain as they are until a PORST.</p> <p><math>00_H</math> The corresponding entry in ETRR can be cleared by setting TRC.</p> <p><math>01_H</math> The corresponding entry in ETRR shall not be cleared by setting TRC.</p>
EOV	15	rh	<p><b>Error Overflow</b></p> <p>The Error Tracking registers have an overflow condition.</p> <p>This bit is preserved until a warm PORST.</p> <p><math>0_B</math> All errors detected since last clear were tracked.</p> <p><math>1_B</math> More errors were detected since last clear than error tracking registers are available. Also, this bit is set if more than one memory block was in error at the same time. See ETRR.MBI for details. The setting of this bit and the forwarding of the overflow error via the UCE alarm is enabled by ECCS.TRE = 1 and ALMSRCS.OVFE = 1. This bit is reset by ECCD.TRC.</p>

Table 478 Reset Values of MCI\_ECCD (i=0-95)

Reset Type	Reset Value	Note
PowerOn Reset	$0000_H$	Warm PORST
Application Reset	-----0 $0000_B$	Application Reset

## Error Tracking Register x

These registers contain the address of errors detected. 5 registers are implemented per SSH instance. ETRR(0) contains the last error detected. Successive errors will push previous errors up. Correctable, uncorrectable and address errors are stored here in the same manner.

The error type (Single Bit Error (SBERR), Double-bit error (DBERR) or Address error (ADDRERR)) at each address entered is stored in the corresponding entry of the **MCI\_ERRINFOx** register. If a new error occurs with different error types (ERRINFO contents will not match) at an address which is already stored, then this address will again be stored for second time in the ETRR registers, this time with the new error types in the corresponding ERRINFO

## Memory Test Unit (MTU)

register. However, a new error at an already stored address, where the ERRINFO contents will match, is not stored again as a separate entry.

Only one new entry can be tracked at one time. If more than one error occurs at the same time in different memory blocks (SRAM towers) ECCD.EOF will be set, no matter how many ETRR registers are still available, and the lowest memory index in which an error occurs is stored in the ETRR.MBI bits. Note that if more than one error occurs at the same time in different sub-towers of same memory block, the ECCD.EOF is not set, and if the error types in the sub-towers are different, all of them are stored in the corresponding same ERRINFO register.

The CERR, and UCERR and the SERR bits in the ECCD register are also set appropriately if enabled.

The address tracked is physical i.e. the address is directly equivalent to the SRAM address input signals.

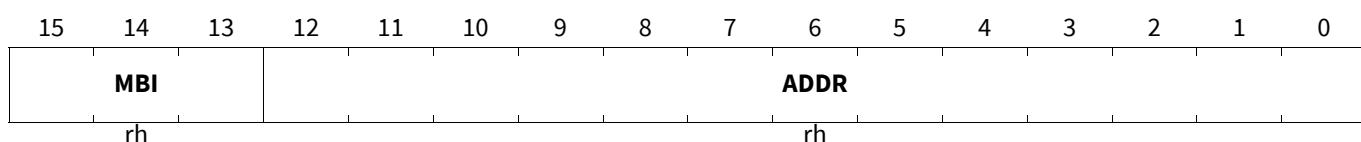
The associated valid bit(s) is/are contained in the register ECCD.VAL. ECCD.TRC clears the associated valid bits and all contents of ETRR and ERRINFO will also be cleared.

It/they contain(s) the faulty address(es) of both the correctable, uncorrectable case and address errors.

Once all registers are used up ECCD.EOF is set.

### **MCi\_ETRRx (i=0-95;x=0-4)**

**Error Tracking Register x**      **( $1012_H + i * 100_H + x * 2$ )**      **PowerOn Reset Value: 0000\_H**



Field	Bits	Type	Description
<b>ADDR</b>	12:0	rh	<b>Address of Error(i)</b> Address of the error detected since last clear operation. If some MSB bits are not required for addressing smaller memories, they are reserved and read as '0'.
<b>MBI</b>	15:13	rh	<b>Memory Block Index of Error(i)</b> If more than one memory is implemented in parallel, these three bits contain the index of the memory block in error to identify the memory in error and the tracked address belongs to this memory. Otherwise these bits always are set to 0.

### **Usage of MCi\_ETRRx registers**

The MCi\_ETRRx registers store the physical addresses -i.e. address at the input of the SRAMs. This address is not the same as system address (shown in MEMMAP chapter). The user should rely on the MCi index (i) to find out in which SSH the error occurred. However it is not recommended to translate the address in MCi\_ETRRx register back to system address.

### **Read Data and Bit Flip Register y**

This register is used for several purposes whenever a register with the size of the memory width is needed.

Normally when test mode is enabled this register contains the data which are directly read back from the RAM (without any data scrambling) during the last read access.

During test mode, it contains the bit flip information. If ECCS.BFLE is set, this information is used to flip bits written to the SRAM.

## Memory Test Unit (MTU)

After a failed test it contains the failed bit map if MCONTROL.FAILDMP is set. The corresponding failure address is contained in the error tracking register.

When MCONTROL.DINIT is set, the content of RDBFL is written to all locations within range.

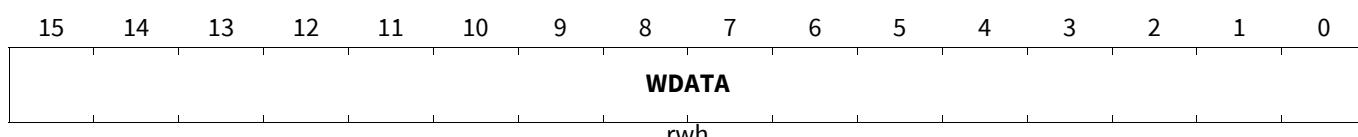
Writes to this register are not permitted while a test is underway.

### MCi\_RDBFLy (i=0-95;y=0-66)

**Read Data and Bit Flip Register y**

( $1060_H + i * 100_H + y * 2$ )

**Application Reset Value: 0000\_H**



Field	Bits	Type	Description
<b>WDATA</b>	15:0	rwh	<b>Word Data</b> This field contains the data of the last memory read operation.

## Alarm Sources Configuration Register

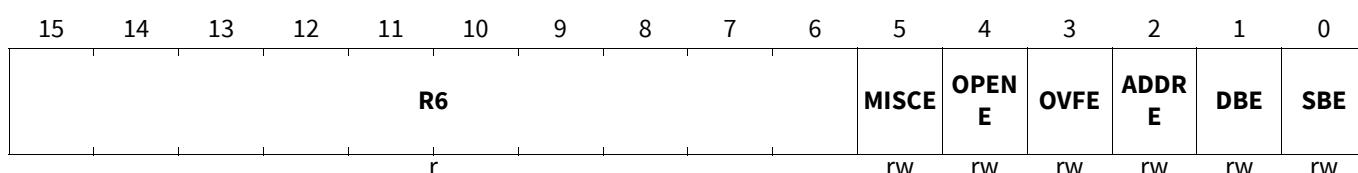
This register controls the internal sources of the 3 SSH alarms (CE, UE and ME). Individual sources can be enabled and disabled, and additionally provides individual status bits for the error/event sources.

### MCi\_ALMSRCS (i=0-95)

**Alarm Sources Configuration Register**

( $10EE_H + i * 100_H$ )

**Application Reset Value: 003F\_H**



Field	Bits	Type	Description
<b>SBE</b>	0	rw	<b>Single Bit Error Notification &amp; Tracking Enable</b> This bit enables ECC Single Bit Detection/Correction event to be tracked forwarded to the CE or UCE alarm. If ECCS.ECE bit is '1', then SBE errors are forwarded to CE alarm. Otherwise to UCE alarm. The error status can be read from the ERRINFO registers (ERRINFO[x].SBERR) 0 <sub>B</sub> SBE errors are neither tracked in the ETRR, nor notified via an alarm. 1 <sub>B</sub> SBE errors are tracked in the ETRR & ERRINFO, and notified via an alarm (CE if ECE = 1, UCE if ECE = 0).
<b>DBE</b>	1	rw	<b>Double Bit Error Notification and Tracking Enable</b> This bit enables ECC Double Bit Errors in the SRAM to be tracked and forwarded as an UCE alarm. The error status can be read from the ERRINFO registers (ERRINFO.DBERR). 0 <sub>B</sub> DBE errors are neither tracked in the ETRR, nor notified via an alarm. 1 <sub>B</sub> DBE errors are tracked in the ETRR & ERRINFO, and notified via a UCE alarm.

**Memory Test Unit (MTU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>ADDRE</b>	2	rw	<p><b>Address Error Notification Enable</b></p> <p>This bit enables the detection and tracking of Address Faults in the SRAM, and forward them as a source of UCE alarm. The error status can be read from the ERRINFO registers (ERRINFO.ADDRERR).</p> <p>0<sub>B</sub> Address Faults in the SRAM are neither tracked in the ETRR, nor notified via an alarm.</p> <p>1<sub>B</sub> Address Faults in the SRAM are tracked in the ETRR &amp; ERRINFO, and notified via a UCE alarm.</p>
<b>OVFE</b>	3	rw	<p><b>ETRR Overflow notification enable- OVFE</b></p> <p>This bit enables the forwarding of the ETRR Overflow event as an alarm source to the UCE alarm. The Error information can be obtained via the ECCD.VALID bits and the EOV bit.</p> <p>0<sub>B</sub> Do not report Error Tracking (ETRR) Buffer Overflow Error.</p> <p>1<sub>B</sub> Report Error Tracking (ETRR) Buffer Overflow Error via the UCE alarm</p>
<b>OPENE</b>	4	rw	<p><b>SSH Operational Error Notification Enable</b></p> <p>This bit enables the forwarding of many errors which are critical to the operation of the SRAM or SSH. These errors are forwarded as one of the sources of the UCE alarm. The error status can be read from FAULTSTS.OPERR bits.</p> <p>This bit is enabled by default.</p> <p>0<sub>B</sub> Do not enable the detection and forwarding SSH/SRAM operation critical errors as a source to the UCE alarm.</p> <p>1<sub>B</sub> Enable the detection and forwarding SSH/SRAM operation critical errors as a source to the UCE alarm</p>
<b>MISCE</b>	5	rw	<p><b>SSH Misc. Errors Notification Enable</b></p> <p>This bit enables the forwarding of many errors which may be critical to the operation of the SRAM or SSH in the future. These errors are forwarded as one of the sources of the ME alarm. The error status can be read from FAULTSTS.MISCERR.</p> <p>This bit is enabled by default.</p> <p>0<sub>B</sub> Do not enable the detection and forwarding of misc. SSH/SRAM errors as a source to the ME alarm.</p> <p>1<sub>B</sub> Enable the detection and forwarding of misc. SSH/SRAM errors as a source to the ME alarm</p>
<b>R6</b>	15:6	r	<p><b>Reserved - Res</b></p> <p>Reads return 0</p>

**SSH Safety Faults Status Register**

This register shows the status of the errors detected in the SRAM, which are forwarded as part of the UCE and ME alarms.

After power-up and before initialization by the SSW the safety flip-flops in the SSH can indicate a fault since some internal registers are not initialized. As a consequence MCI\_FAULTSTS.MISCERR[2] could be set and result in an alarm. LBIST does initialize the internal registers and clears the error. Alarms resulting from MCI\_FAULTSTS.MISCERR[2] should be ignored during start-up and cleared right after execution of the SSW in case LBIST was not run.

## Memory Test Unit (MTU)

## MCi\_FAULTSTS (i=0-95)

## SSH Safety Faults Status Register

(10F0<sub>H</sub>+i\*100<sub>H</sub>)PowerOn Reset Value: 0000<sub>H</sub>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R14				MISCERR				R6				OPERR			

r                            rwh                            r                            rwh

Field	Bits	Type	Description
OPERR	5:0	rwh	<p><b>SSH Critical Operation Error Occured</b>            One bit status corresponding to each of the error sources contributing to the Critical operational error sources to the Un-Correctable Error alarm (UCE). Enabled by ALMSRCS.OPENE. If multiple errors happened, multiple bits are set at the same time. To clear, write '0'. Write of '1' has no effect. Even if any bit is set, further errors are still forwarded.            Unspecified bits are reserved for future use and shall always return 0.</p> <ul style="list-style-type: none"> <li>01<sub>H</sub> SSH has been enabled. Functional access to SRAM is disabled.</li> <li>02<sub>H</sub> Auto-data-init or Partial-erase has been triggered. Part or whole of the SRAM may be overwritten.</li> <li>04<sub>H</sub> An error has been detected by safety Flip-Flops in one of the registers in the SSH. This bit is also always set at the end of a safety Flip-Flop self test (triggered by setting ECCS.SFFD) since the error paths are tested - but a real fail in the test is only indicated if MISCERR[0] is also set.</li> <li>08<sub>H</sub> Unexpected triggering of MBIST FSM, or Test access to SRAM, or Test features (eg. muxes) in the data path, leading to potential data corruption.            Set when CONFIG1.AG_MOD is switched to non-zero value or MCONTROL.START is set to trigger a test.</li> </ul>
R6	7:6	r	<p><b>Reserved - Res</b>            Reads return 0</p>
MISCERR	13:8	rwh	<p><b>SSH Miscellaneous Error Status- MISCERR</b>            One bit status corresponding to each of the error sources contributing to the Miscellaneous Error (ME) alarm. Enabled by ALMSRCS.MISCE. If multiple errors happened, multiple bits are set at the same time. To clear, write '0'. Write of '1' has no effect. Even if any bit is set, further errors are still forwarded. Unspecified bits are reserved for future use and shall always return 0.</p> <ul style="list-style-type: none"> <li>01<sub>H</sub> Failure detected during safety Flip-Flop self test (triggered by setting ECCS.SFFD). The ME alarm is not triggered by this fail. Hence the software shall poll the status of this bit to check if the safety Flip-Flop self test failed.</li> <li>02<sub>H</sub> Alarm notification disabling detected. Any of the alarms may not be forwarded in the future.</li> <li>04<sub>H</sub> Safety mechanism disabling detected. Some of the SRAM or SSH related errors may not be detected in the future.</li> <li>08<sub>H</sub> Write zero, read as zero. Reserved for future use.</li> </ul>

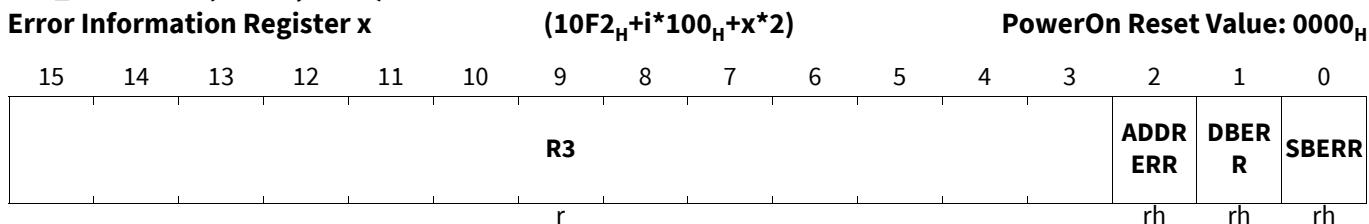
## Memory Test Unit (MTU)

Field	Bits	Type	Description
R14	15:14	r	<b>Reserved - Res</b> Reads return 0

### Error Information Register x

The **MCI\_ETRRx** register(s) contain(s) the address of the error(s) detected. The type of error at the memory location corresponding to each ETRR address entry can be found using the Error Information Registers (ERRINFO). One ERRINFO register is implemented corresponding to each ETRR entry. An ERRINFO register is valid only when the corresponding ETRR entry is valid (ECCD.VAL = 1), and is cleared by ECCD.TRC.

#### MCI\_ERRINFOx (i=0-95;x=0-4)



Field	Bits	Type	Description
<b>SBERR</b>	0	rh	<b>Single Bit Error Detected</b> Read as: 0 <sub>B</sub> No Single Bit error detected at the memory address in the corresponding ETRRx register. 1 <sub>B</sub> Single Bit detected at the memory address in the corresponding ETRRx register.
<b>DBERR</b>	1	rh	<b>Double Bit Error Detected</b> Read as: 0 <sub>B</sub> No Double-Bit error detected at the memory address in the corresponding ETRRx register. 1 <sub>B</sub> Double Bit error detected at the memory address in the corresponding ETRRx register. Note that for SRAMs with Address-ECC (refer to the Appendix chapter for the list of such SRAMs in the device), this bit is also set if an error in the Address bits are detected.
<b>ADDRERR</b>	2	rh	<b>Address Fault Detected</b> Read as: 0 <sub>B</sub> No address error detected at the memory address in the corresponding ETRRx register. 1 <sub>B</sub> Address error detected at the memory address in the corresponding ETRRx register. Note that for SRAMs with Address-ECC, this bit is not used. For such SRAMs, errors in both Address and Data bits are notified by the DBERR bits.
<b>R3</b>	15:3	r	<b>Reserved - Res</b> Reads return 0

## Memory Test Unit (MTU)

### 13.5 Safety Measures

Please refer to the safety manual as the final reference regarding safety mechanisms and reactions. The following chapter only aims to provide a generic description.

There are a number of safety features that help to detect random hardware faults in the SRAM cells as well as periphery.

Random hardware faults in the data (data path as well as SRAM cell), address logic within the SRAM and control logic within the SRAM are protected by safety mechanisms.

Safety notifications (alarms) are provided to the SMU.

This section provides a short summary of these safety mechanisms and safety notifications. Additionally, some of these features may be explained in more detail in other sections.

#### 13.5.1 Safety Features

The safety features implemented in the SRAMs, SSH instances and the MTU are the following:

- Error Detection and Correction Codes / Logic
- Address Error Monitor
- SRAM Mux Factor
- Error Tracking Registers
- Safety Flip-Flop implementation for critical register bits.

##### 13.5.1.1 SRAM Error Detection & Correction (EDC/ECC)

All SRAMs are implemented with Error Detection or Correction Codes (ECC) for the stored Data. Except for certain SRAMs described in the appendix chapter, the ECC is computed over the data alone.

During a WRITE operation, checksum bits (ECC bits) for the input data word are generated using an ECC encoder. These bits are stored in the SRAM along with and additionally to the data word itself.

During a READ operation, an ECC decoder compares the read Data word + ECC bits to the expected ECC value. If due to any random hardware failure, soft error etc., the data word stored in the SRAM was corrupted, then there will be a mismatch compared to the expected ECC value. This is notified as an error and alarm.

Two different kinds of ECC codes are used in the system for different SRAMs - SECDED codes or DED codes.

SECDED codes can detect upto a double bit error (DBE), as well as correct a single bit error (SBE). That is, the Hamming distance of these codes = 4.

DED codes can detect upto a double bit error, but cannot correct any error.

The column "ECC Type" in the SSH instances table shows which type of ECC code is used for each SRAM.

There are two error notifications sent to the SMU from the ECC decoder via the CE and UCE alarms. For SRAMs with SECDED ECC- correctable error event (ie SBE corrected or detected) alarm and Un-correctable error detected (ie DBE detected) alarm. For SRAMs with DED ECC, there is only one error notification and this is notified by the DBE error and UCE alarm.

SBE notification (Depending on ECCS.ECE, forwarded as a source in CE alarm or UCE alarm) as well as tracking in the ETRR/ERRINFO is enabled by setting ALMSRCS.SBE = 1.

DBE notification (Forwarded as a source in UCE alarm) as well as tracking in the ETRR/ERRINFO is enabled by setting ALMSRCS.DB = 1.

Please note that the ECCS.CENE, UCENE and TRE bits act as a further global enable and have to be set for the ECC alarm notifications to be sent to SMU as well as to enable error tracking in the ETRR/ERRINFO.

The number of ECC words in an SRAM tower is defined by the ECC granularity, described in the SSH instances table in the appendix chapter.

## Memory Test Unit (MTU)

For example: An ECC granularity of 2 implies that there are 2 ECC words per physical SRAM block.

### 13.5.1.2 Address Error Monitor

Similar to how the ECC helps to monitor errors in the data, the Address Error Monitoring mechanism helps to detect failures in the address generation logic within the SRAMs.

An error in the address generation within the SRAM can lead to:

- A valid input address selecting no wordline/bitline
- A valid input address selecting multiple wordlines/bitlines
- A valid input address selecting a wrong wordline/bitline
- A valid input address selecting a non-existing (eg. out of range) wordline/bitline

All the above SRAM failure modes are detected by the address error monitor.

Whenever an error is detected by the address error monitor, an alarm is sent to the SMU.

The address error detection, tracking and notification is enabled by setting ALMSRCS.ADDRE, and the alarm is forwarded as one of the sources to the UCE alarm.

If an address error is enabled (ALMSRCS.ADDRE = 1) and an error is detected, then it is tracked in the ETRR and ERRINFO registers (ERRINFO.ADDRERR is set). Please note that the ECCS.UCENE and TRE bits act as a further global enable and have to be set for the address error alarm notifications to be sent to SMU as well as to enable error tracking in the ETRR/ERRINFO.

**NOTE:** In devices with EMEM, EMEM SRAM does not have address signatures. Instead, the ECC is computed out of both address and data. These SRAMs are described in the appendix chapter. Hence in these SRAMs, the Address Error detection is enabled via ALMSRCS.DBE itself, and the status notified in ERRINFOx.DBERR. The ALMSRCS.ADDRE bit has no effect in this case.

### 13.5.1.3 SRAM Mux Factor

In order to reduce the probability of a single random hardware fault (e.g. an alpha particle strike or contact fail) causing more than a single-bit error, the bits in each logical data word are interleaved by a certain factor (4, 8 or 16). This means, two consecutive bits in a logical data word are actually physically 4, 8 or 16 bits apart. The mux factor used in each SRAM is mentioned in the SSH instances table.

### 13.5.1.4 Error Tracking Registers

The error tracking registers store the addresses (ETRR) and the corresponding type (ERRINFO) of errors in the SRAM. When a threshold of errors is reached (i.e. when the ETRR registers are fully filled) - then an overflow alarm is raised to the SMU. The system can react appropriately to such an overflow.

Error Tracking is globally enabled by setting ECCS.TRE. Tracking of individual error types (SBE, DBE, ADDRE) can be enabled or disabled via the corresponding bits in the ALMSRCS register.

Additionally, ALMSRCS.OVFE enables the forwarding of overflow error notification via the UCE alarm to the SMU.

#### Error Tracking

It is possible for the software to track the error type and the address in which it occurred. The ECCS.TRE enables the tracking of the address and type of a detected error.

Whenever address tracking is enabled, the 5 ETRRx(x = 0-4) (Refer [MCI\\_ETRRx \(i=0-95;x=0-4\)](#)) registers contain the physical address of the detected error. ETRR0 contains the last error detected. The ECCD.VAL is a 5 bit field corresponding to the 5 ETRR registers. When any of the ETRRx registers contain a valid error address, the corresponding ECCD.VAL bit is also set. The [MCI\\_ERRINFOx \(i=0-95;x=0-4\)](#) registers contain the error type (Correctable, Uncorrectable or Address Error) at the address in the corresponding ETRRx register entry.

## **Memory Test Unit (MTU)**

Software can clear the ECCD.EOF, ECCD.VAL and the ETRRx(x=0-4) and ERRINFOx(x=0-4) registers by setting the tracking clear bit, ECCD.TRC. The errors are always tracked since the last clear.

### **Soft Error Handling**

The ECCD.PERMERR bits are useful to separate permanent and soft errors, if required. When an error occurs, software may probe the location to ascertain if it is a hard error or not.

When PERMERR[x] is set by software, then the corresponding ETRR/ERRINFO entry is denoted to be a permanent error, and are hence no longer considered part of the error tracking buffers. Even setting ECCS.TRC has no effect in clearing such entries. In such a case, even the overflow is generated when all the entries with PERMERR = 0 are filled.

If all entries are marked with PERMERR[x] = 1, then any new error automatically triggers the overflow.

The PERMERR bits shall be set by software only if the corresponding VALID bits are already set. The PERMERR bits shall not be set when the corresponding VALID bits are 0.

### **13.5.1.5 Safety Flip-Flops**

Certain critical registers have to be protected against soft errors changing their values. These registers protected by safety FF are:

In the MTU:

- MTU\_MEMMAP (SFF Detection Only)

In each SSH:

- ECCS: MENE, ECCMAP, SFLE & BFLE bits protected by DED ECC SFFs - UE Error notifications OR-ed and sent via UCE alarm & OPERR[2].
- ALMSRCS: MISCE bit protected by ECC DED SFF - Error notification via UCE alarm & OPERR[2].

The safety FFs in the MTU report errors directly to the SMU. The safety FFs in the SSH report errors via the UCE alarm and the status via OPERR[2].

### **13.5.2 Safety Notifications**

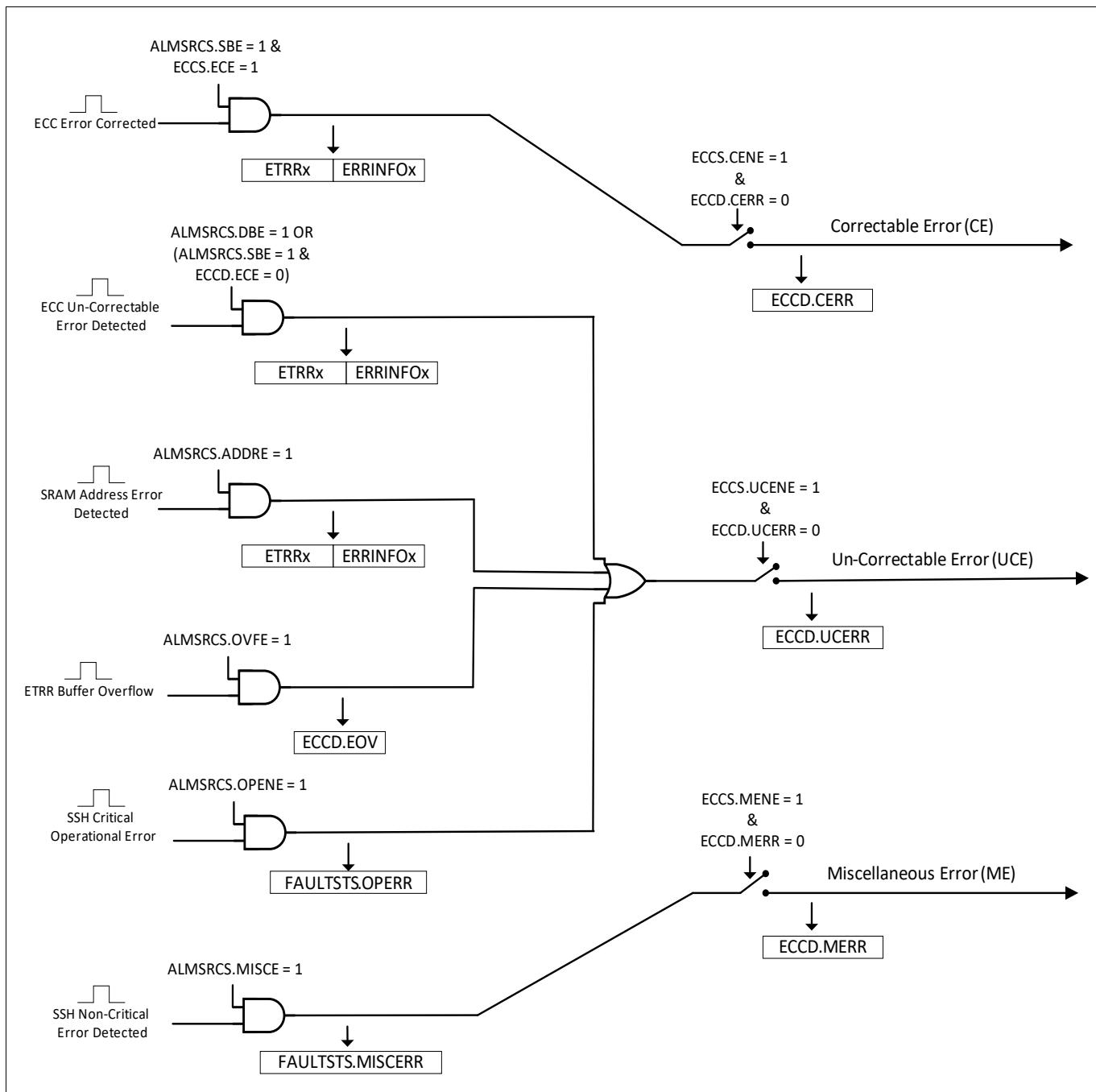
The error notifications from all SRAMs/SSHS in the system can be reported to the Safety Management Unit (SMU). The SMU may be programmed to initiate appropriate action.

Each SSH provides the following alarms to the MTU which are then forwarded to the SMU:

- SRAM ECC Correction Event Occurred Alarm (CE alarm)- Status via ECCD.CERR bit.
- SSH Un-Correctable Event Critical for Operation Alarm (UCE alarm)- Status via ECCD.UCERR bit.
- SSH Miscellaneous Error Events Alarm (ME alarm) - Status via ECCD.MERR bit.

These alarms are forwarded from each SSH to the MTU. The MTU consolidates the alarms from all the SSHs and forwards this further to the SMU.

## Memory Test Unit (MTU)



**Figure 155 Alarms and Error Detection enabling (ECCS.TRE assumed to be set).**

When an alarm occurs, the corresponding \*ERR bit in the ECCD register is set. No new alarms of the same type are forwarded as long as this bit remains set. Hence, once an alarm occurs, the software shall clear the corresponding \*ERR bit in order to enable forwarding of further alarms of the same type again.

Normally, there may be a time window of multiple clock cycles from when an alarm event occurs (ie ECCD.\*ERR is set) to until the software reacts and clears the flag to enable forwarding further alarms. During this time window, further new errors corresponding to the same alarm type may occur. For any such new event, although no new alarm is generated, the errors are still tracked - i.e. - the information about new errors are stored in the ETRR, ERRINFO, and FAULTSTS registers. Thus software may retrieve the information regarding new errors during this window from these status registers.

## Memory Test Unit (MTU)

Even if the new hardware event happens exactly at the same cycle as the software clear event takes effect - the ECCD bits are cleared and a new alarm is not generated. Hence software has to always rely on the ETRR, ERRINFO and FAULTSTS registers for new errors occurring during the time window of clearing.

NOTE: As long as an alarm notification is enabled, and the corresponding \*ERR bit is not set, it shall be ensured that no alarm is blocked, lost or cleared.

**Table 479 SSH Safety Notifications Summary**

Alarm	Errors or Events Mapped	Enable	Software Handling / Clearing
CE	- ECC Correction Event Occurred	ECCS.CENE = 1 and ECCS.ECE = 1 and ALMSRCS.SBE = 1	ECCD.CERR has to be cleared to forward further CE alarms. The error is always tracked in the ETRR & ERRINFO registers independent of the value of CERR. If ECE is '0', All enabled ECC errors are notified to the UCE alarm.
UCE	- ECC Uncorrectable Event Detected - Address Error Detected - ETRR Overflow Occurred - Error Critical for SSH Operation Detected (Incl. Critical Safety-FF Errors)	- ECCS.UCENE = 1 and - ALMSRCS.ADDRE = 1, or - ALMSRCS.DBE = 1 or - ECCS.TRE = 1, ALMSRCS.OVFE = 1 and event to set EOF, or - ECCS.ECE = 0 and ALMSRCS.SBE = 1, or - ALMSRCS.OPENE = 1	ECCD.UCERR has to be cleared to forward further UCE alarms. The errors and status are always tracked in the ETRR & ERRINFO & ALMSRCS registers independent of the value of UCERR.
ME	- Less Critical Safety-FF Errors (Enable Bits in ECCS, ALMSRCS) - Disabling of alarm notifications. - Disabling of safety mechanisms.	ECCS.MENE = 1 and - ALMSRCS.MISCE = 1	ECCD.MERR has to be cleared to forward further ME alarms.

### 13.5.2.1 Alarm Handling

As shown in **Table 479**, there are 3 alarms from each SSH, and different events / errors are mapped to each alarm. Each alarm is enabled by the corresponding enable bit in the ECCS register (i.e. ECCS.CENE, UCENE and MENE). When an alarm occurs, the corresponding status flag in the ECCD register is set (i.e. ECCD.CERR; UCERR and MERR).

For the events mapped to each alarm type, there are separate enable bits in the ALMSRCS register (all enabled by default).

For ECC SBE or DBE error event and Address Faults (ADDRE) - the status of the error can be obtained from the ETRR and ERRINFO registers (and ECCD.VALID bits).

Similarly, Error tracking overflow is enabled by setting ALMSRC.OVFE, and an overflow occurs, then the ECCD.EOF bit is set. Of course, the tracking itself happens only if ECCS.TRE = 1 (by default).

For safety reasons, some operational faults are detected which affect the normal operation or access to the SRAM (e.g. SSH gets accidentally enabled, Auto-data-init gets triggered etc). These are also notified via the UCE alarm, and can be enabled via the ALMSRCS register OPENE bit (Operational Error Notification Enable). Please refer to

## Memory Test Unit (MTU)

**“Mapping of Errors to FAULTSTS.OPERR and UCE alarm” on Page 47 and “Mapping of Errors to FAULTSTS.MISCERR and ME alarm” on Page 48** for the error mapping.

**Attention:** *The MTU clock has to be enabled via the MTU\_CLC register to forward any SSH alarm.*

Certain faults do not directly impact the usage of the SRAM, but if left latent, could potentially have a serious effect- for example, the ECC or Address Error notification enable bits or the Alarm enable bits get flipped due to a soft error. Such errors are enabled by the ALMSRCS.MISCE bit.

### 13.5.2.1.1 Alarms after startup

The System Firmware of the AURIX™ TC3xx platform performs certain operations on the SSH, such as configured SRAM initialization (via PROCONRAM register) and timing and redundancy installation. Since these operations involve enabling SSHs and writing to the SRAM, they can trigger SSH alarms, specifically the UCE alarm via the FAULTSTS.OPERR[0] and OPERR[3]. The corresponding alarm status bits are left uncleared as an indication of correct System Firmware execution.

- After any System Reset: For each and every SSH in the system, the UCE alarm status in the SMU, the ECCD.UCERR (Consequently also SERR) and the FAULTSTS.OPERR[0] will be set.
- For SRAMs enabled for initialization via firmware (via PROCONRAM) - additionally the FAULTSTS.OPERR[3] flag will be set after the reset (Warm or Cold Porst) on which SRAM initialization is enabled.
- If the above flags are set, then clear these flags and continue with normal application startup and starup safety checks. It may be assumed that the boot was safely completed with regards to SRAM handling and initialization.
- If any of the flags are not set, it shall be assumed that the System Firmware boot did not safely complete with regards to SRAM handling and initialization. To proceed from such a situation, the application shall follow the recommendations in the safety manual.

**Attention:** *In case of an application reset, there are no SSH alarms or status bits triggered by the system firmware. Hence any alarm or error status after an application reset serves as an indication of a real error.*

**Note:** *The above sequence also applies to the standby controller SRAMs (SCR XRAM & IRAM) - but needs to be taken into account only after a cold-power-on reset and SCR was not enabled; instead of a system reset.*

**Note:** *Special case of EMEM: In case of an emulation device, due to the PROLOG code handling by the System Firmware, the FAULTSTS.MISCERR[1] and [2] will be additionally set in the case of a cold PORST and Warm PORST*

### 13.5.2.1.2 Diagnostics

When an alarm from an SSH occurs, the software has to check multiple registers to diagnose the causes of the fault.

ECCD.CERR, UCERR and MERR bits give the overall status of which alarm occurred.

ETRR/ERRINFO registers store the address and the fault type of a fault in the SRAM (i.e. SBE, DBE or ADDR error).

FAULTSTS register bits store the status of other faults in the SSH itself which are mapped to UCE or ME alarm.

All the error status bits (ETRR, ERRINFO, FAULTSTS, ECCD.VAL, PERMERR & EOV) are preserved until a power-on reset. The alarm status (ECCD.CERR, UCERR, MERR and SERR) are cleared on an application reset.

Software can clear the bits by writing to them (when specified) or by setting the TRC bit as appropriate.

Please refer to **Table 480** and **Table 481** for hints towards runtime error diagnosis and application recovery.

## Memory Test Unit (MTU)

### 13.5.2.1.3 Error Mapping

Both the UCE and the ME alarms have multiple events mapped to them. These events are tapped in such a way that the logic cone that can potentially cause the fault is covered as much as possible.

The OPERR bit in the FAULTSTS register stores the status of various errors critical to SSH or SRAM operation.

**Table 480 Mapping of Errors to FAULTSTS.OPERR and UCE alarm**

Error Status Bit	Error Description during normal application run	Hints for software reaction for recovery
FAULTSTS.OPERR[0]	SSH enabled. Functional access to SRAM is disabled.	Perform an application reset <sup>1)</sup> .
FAULTSTS.OPERR[1]	Auto-data-init or Partial erase (caches) was triggered, resulting in overwriting of SRAM data.	Perform an application reset <sup>2)</sup> .
FAULTSTS.OPERR[2]	Safety Flip-Flop error detected in one of the registers. (After a self test triggered by SFFD, this error has to be taken into account together with MISCERR[0]).	Perform a system reset. - Initialize the SRAM & SSH registers - Trigger a safety flip-flop self test via ECCS.SFFD. - Check that the test does not report a fail (MISCERR[0]).  If the error still persists and cannot be cleared, a warm PORST has to be issued before performing the initialization and safety flip-flop self test.
FAULTSTS.OPERR[3]	MBIST FSM got triggered, test muxes got enabled in the data path, or some other random hardware failure triggered caused data in the SRAM to be overwritten or corrupted by a part of SSH logic. Set when CONFIG1.AG_MOD is switched to non-zero value or MCONTROL.START is set to trigger a test.	Perform an application reset <sup>1)</sup> .

1) For FSI, perform a system reset. For SCR RAMs, a cold PORST is required.

2) For FSI, a system reset is required.

The MISCERR bit in the FAULTSTS register stores the status of various errors which may not be directly critical for SSH or SRAM operation, but if left latent, may result in a critical failure in the future.

## Memory Test Unit (MTU)

**Table 481 Mapping of Errors to FAULTSTS.MISCERR and ME alarm**

Error Status Bit	Error Description during normal application run	Hints for software reaction for recovery
FAULTSTS.MISCERR[0]	This bit can be set only after the software has triggered a Safety Flip-Flop self test via the ECCS.SFFD bit. If the test detected an error and the test failed, this bit is set. (The ME alarm is not triggered in this case).	<p>Perform a system reset.</p> <ul style="list-style-type: none"> <li>- Initialize the SRAM &amp; SSH registers</li> <li>- Trigger a safety flip-flop self test via ECCS.SFFD.</li> <li>- Check that the test does not report a fail (MISCERR[0]).</li> </ul> <p>If the error still persists and cannot be cleared, a warm PORST has to be issued before performing the initialization and safety flip-flop self test.</p>
FAULTSTS.MISCERR[1]	Any of the alarm notifications of UCENE or CENE got disabled (i.e. one or more of the bits CENE or UCENE in the ECCS register went from 1 to 0). Note: ECCS.MENE is protected by Safety Flip Flop.	Re-initialize SSH registers (ECCS).
FAULTSTS.MISCERR[2]	Any of the safety mechanisms got disabled (i.e. ECE, TRE, SBE, DBE, ADDRE, OVFE, OPENE) was set from 1 to 0. Note: ALMSRCS.MISCE is protected by a Safety Flip Flop.	Re-initialize SSH registers (ECCD, ALMSRCS).

**13.5.2.1.4 Error Injection and Alarm Triggering**

To ensure proper behaviour of safety related software, it may be required to inject errors and trigger different safety notifications.

In order to trigger ECC errors (status denoted by ERRINFO.SBERR and DBERR), the ECCMAP bits may be used. When ECCMAP is not equal to 0, then either data or ECC bits alone can be accessed. Using this feature, single or double bit soft errors may be introduced in the SRAM.

Note that ECCMAP is intended for error injection before a test, and shall not be set to non-zero during a test itself.

The Address errors (status denoted by ERRINFO.ADDRERR) may be triggered by setting the SFLE bit, which can force an error into the address by flipping one address bit.

The CE and UCE alarms may be triggered by injecting an ECC or address error.

In order to trigger an ME alarm for test purpose, any of the safety notifications can be simply disabled. This will trigger an ME alarm.

**Memory Test Unit (MTU)****13.6 Revision History****Table 482 Revision History**

Reference	Change to Previous Version	Comment
<b>V7.4.7</b>		
<a href="#">Page 49</a>	Revision History entries up to V7.4.6 removed.	
<a href="#">Page 41</a>	Added sentence at the end of “SRAM Error Detection & Correction (EDC/ECC)” section.	
<a href="#">Page 20</a>	Corrected footnote rendering for MEMTEST in User Manual	
<b>V7.4.8</b>		
<a href="#">Page 49</a>	Revision History entries up to V7.4.7 removed.	
<a href="#">Page 38</a>	Added paragraph to “SSH Safety Faults Status Register” section referring to alarms resulting from MCiFAULTSTS.MISCERR[2] to be ignored during start-up and cleared right after execution of the SSW in case LBIST was not run.	
<b>V7.4.9</b>		
<a href="#">Page 21</a>	Corrected footnote rendering for MEMMAP in User Manual.	
<a href="#">Page 49</a>	Removed two unnecessary entries in revision history for V7.4.8.	
<a href="#">Page 11</a>	Added statement “Additional memory test algorithms are supported by the SSH, but used in production test modes only.”	
<b>V7.4.10</b>		
-	No functional changes.	
<b>V7.4.11</b>		
<a href="#">Page 11</a>	Updated text in Section <b>Non-Destructive Test (NDT)</b> .	
<a href="#">Page 42</a>	Updated Note in Section <b>Address Error Monitor</b> .	
<a href="#">Page 20</a>	Updated Register <b>Memory MBIST Enable Register i</b> .	

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

### 14 General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Each Port module “Port slice” controls a set of assigned General Purpose Input/Output (GPIO) port lines which are connected to pads connected to device pins/balls.

#### 14.1 Feature List

Depending on its configuration a Port module can have the following features:

- Controls up to 16 port lines.
- Enables SW to control the output of each port line.
- Output modification registers ease clearing, setting and toggling of single port lines and nibbles of port lines without affecting the state of other port lines.
- Enables SW to read the input value of each port line.
- Multiplexes up to 7 alternate functions to each output.
- Supports direct I/O control by a peripheral on a per line granularity.
- Controls pad characteristics of the assigned pads like drive strength, slew rate, pull-up/down, push/pull or open-drain operation, selection of TTL or CMOS/automotive input levels.
- The emergency stop feature allows to switch off the output driver of configurable port lines by SMU or special port pins.
- For pad pairs with LVDS functionality it controls LVDS characteristics and allows switching between LVDS and CMOS modes.
- In packages with reduced pin count the Port module can disable selected pins.

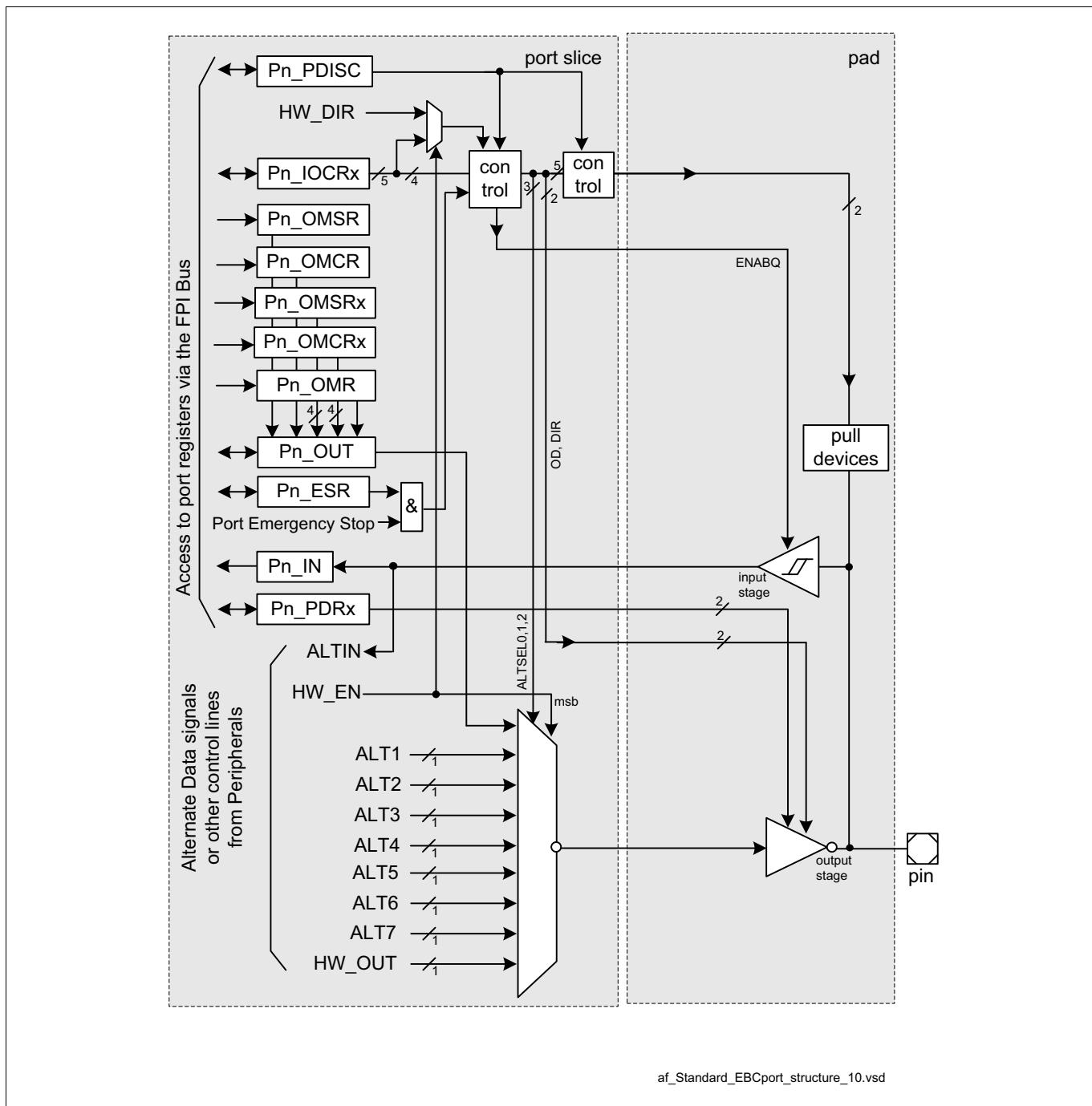
#### Device Dependent Implementation of Ports

The implementation of the Ports is dependent on the device. The configuration is partly done during startup by the Firmware, e.g. to disable pins not available in a certain package. The device specific addendum of this specification describes the configuration.

#### 14.2 Overview

[Figure 156](#) is a general block diagram of a GPIO port slice not showing LVDS functionality.

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)



**Figure 156 General Structure of a Port Pin**

Each port line has a number of control and data bits, enabling very flexible usage of the line. Each port pad can be configured for input or output operation. In input mode (default after reset), the output driver is switched off (high-impedance). The actual voltage level present at the port pin is translated into a logical 0 or 1 via a Schmitt-Trigger device and can be read via the read-only register Pn\_IN. Input signals are connected directly to the various inputs of the peripheral units. The function of the input line from the pin to the input register Pn\_IN and to the peripheral is independent of whether the port pin operates as input or output. This means that when the port is in output mode, the level of the pin can be read by software via Pn\_IN or a peripheral can use the pin level as an input.

In output mode, the output driver is activated and drives the value supplied through the multiplexer to the port pin. Switching between input and output mode is accomplished through the Pn\_IOCR register, which enables or

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

disables the output driver. If a peripheral unit uses a GPIO port line as a bi-directional I/O line, register Pn\_IOCR has to be written for input or output selection. The Pn\_IOCR register further controls the driver type of the output driver, and determines whether an internal weak pull-up, pull-down, or without input pull device is alternatively connected to the pin when used as an input. This offers additional advantages in an application.

The output multiplexer in front of the output driver selects the signal source for the GPIO line when used as output. If the pin is used as general-purpose output, the multiplexer is switched by software (Pn\_IOCR register) to the Output Data Register Pn\_OUT. Software can set or clear the bit in Pn\_OUT through separate Pn\_OMSR or Pn\_OMCR registers. The set or clear operations for the bits in Pn\_OUT can also be done for up to four bits per register in Pn\_OMSRx and Pn\_OMCRx ( $x=0,4,8,12$ ). Alternatively, the set, clear or toggle function can be achieved through Pn\_OMR, where adjacent pins within the same port can be set, cleared or toggled within one write operation. The manipulation of the control bits in these registers can directly influence the state of the port pin. If the on-chip peripheral units use the pin for output signals, the alternate output lines ALT1 to ALT7 can be switched via the multiplexer to the output driver. The data written into the output register Pn\_OUT by software can be used as input data to an on-chip peripheral. This enables, for example, peripheral tests via software without external circuitry.

When selected as general-purpose output line, the logic state of each port pin can be changed individually by programming the pin-related bits in the Output Modification Set Register Pn\_OMSR, Output Modification Set Register x Pn\_OMSRx ( $x=0,4,8,12$ ), Output Modification Clear Register Pn\_OMCR, Output Modification Clear Register x Pn\_OMCRx ( $x=0,4,8,12$ ) or Output Modification Register, OMR. The bits in Pn\_OMSR/Pn\_OMSRx and Pn\_OMCR/Pn\_OMCRx make it possible to set and clear the bits in the Pn\_OUT register. While the bits in Pn\_OMR allows the bits in Pn\_OUT to be set, cleared, toggled or remain unchanged.

When selected as general-purpose output line, the actual logic level at the pin can be examined through reading Pn\_IN and compared against the applied output level (either applied through software via the output register Pn\_OUT, or via an alternate output function of a peripheral unit). This can be used to detect some electrical failures at the pin caused through external circuitry. In addition, software-supported arbitration schemes can be implemented in this way using the open-drain configuration and an external wired-And circuitry. Collisions on the external communication lines can be detected when a high level (1) is output, but a low level (0) is seen when reading the pin value via the input register Pn\_IN.

Most of the digital GPIO lines have an emergency stop logic<sup>1)</sup>. This logic makes it possible to individually disconnect outputs and put them onto a well defined logic state in an emergency case. In an emergency case, the pin is switched to input function with internal pull-up device connected or tri-state (depending on global configuration). The Emergency Stop Register Pn\_ESR determines whether an output is enabled or disabled in an emergency case.

### LVDS Port Operation

The LVDS pads offer LVDS RX or TX functionality for a pair of pins (see Pinning). Additionally they contain — if not specified differently — a bidirectional CMOS pad for each of the pins. The CMOS functionality is controlled by above described set of standard registers. The LVDS functionality is controlled by additional Pn\_LPCR registers ([Chapter 14.4.4](#)).

## 14.3 Functional Description

The following description add further details to [Chapter 14.2](#).

### 14.3.1 System Connectivity of Ports

The connectivity of the Ports is described in the pinning tables. These are contained in the Data Sheet.

1) This feature is not available for all port lines, for details see Data Sheet.

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

**Table 483 Example Port Table**

Pin	Symbol	Ctrl.	Buffer Type	Function	
10	Pxx.y	I	FAST / PU1 / VEXT	<b>General-purpose input</b>	
	TIMm_n			<b>GTM_TIN</b>	
	Pxx.y	O0		<b>General-purpose output</b>	
	TOMa_b	O1		<b>GTM_TOUT</b>	
	TOMc_d			<b>GTM_TOUT</b>	
	IOM_REFv_w	O2		<b>IOM reference input</b>	
	ASCLINz RTS			<b>ASCLIN0 output</b>	
	...	...		...	
	ETH0_MDIO	O		<b>Ethernet output</b>	

The symbols in the **Ctrl.** column indicate the multiplexer connectivity:

I = Input (for GPIO port lines with IOCR bit field Selection  $PCx = 0XXX_B$ )

AI = Analog input

O = Output (the assigned function is active when a peripheral takes control of the port line)

O0 = Output with IOCR bit field selection  $PCx = 1X000_B$  (Pxx.y)

O1 = Output with IOCR bit field selection  $PCx = 1X001_B$  (ALT1)

O2 = Output with IOCR bit field selection  $PCx = 1X010_B$  (ALT2)

O3 = Output with IOCR bit field selection  $PCx = 1X011_B$  (ALT3)

O4 = Output with IOCR bit field selection  $PCx = 1X100_B$  (ALT4)

O5 = Output with IOCR bit field selection  $PCx = 1X101_B$  (ALT5)

O6 = Output with IOCR bit field selection  $PCx = 1X110_B$  (ALT6)

O7 = Output with IOCR bit field selection  $PCx = 1X111_B$  (ALT7)

To each input several functions can be connected. The peripherals' configuration defines if this input is used.

The port module decides which of the 8 output signals O0 to O7 drives the pad.

Some Ox rows list more than one function, e.g. several TOM outputs and IOM reference inputs. The GTM module (see corresponding chapter) has its own sub-multiplexer structure that defines which of the GTM sub-units drives this signal. Additionally the IOM modules "listens" on these output signals (see IOM chapter).

### Pins without Port module

The pinning of some devices shows Px.y pins without assigned Port module "Px". This is the case for RIF module pins (P50.y and P51.y) for which the RIF module itself contains the necessary control registers.

### Control of Port Line by Peripheral (HW\_DIR/HW\_EN/HW\_OUT)

By using the HW\_DIR/HW\_EN/HW\_OUT interface (see [Figure 156](#)) a peripheral can take over control of the output path of a port line. When the peripheral activates HW\_EN its data line connected to HW\_OUT is driven to the output driver. With HW\_DIR the peripheral can control the output driver activation. The data line connected to HW\_OUT is shown in the "O" rows in the pinning tables.

When a peripheral activates HW\_DIR and HW\_EN the output driver operates in push/pull mode.

**Note:** *The SMU uses for FSP1 this interface therefore FSP1 operates in push/pull mode. For FSP0 the SMU takes direct control of the output pad by copying the configuration of this pad from the Port module. Therefore FSP0 can operate also in open-drain mode. For details of this cooperation see SMU chapter.*

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

### 14.4 Registers

The following describes the register set of a generic fully featured Port module.

**Note:** *Destructive read is not implemented in any of the registers.*

**Note:** *Parallel requests from on chip bus masters to the ports module are executed sequentially via the on chip bus system. Read-modify-write feature provides an atomic read/write sequence where no other master can access the ports module in between the operations*

#### Implementation

The implemented port modules are derived from this generic module. Registers for unavailable functionality are usually not implemented. **Table 484** describes general rules for realization of registers with unavailable Port lines. The exact description of each register can be found in the device specific appendix.

**Table 484 Register Behavior of Unavailable Port Lines**

Register	Behavior
Pn_OUT, Pn_ESR, Pn_PDR0, Pn_PDR1, Pn_IOCRx (x=0-3), Pn_PDISC	The unused control fields within the same nibble pin group as available pins are "rw", returns value that was last written and must be written with reset value. Otherwise, these unused control fields are 'r', read as 0 and should be written with reset value.
Pn_OMR, Pn_OMSR, Pn_OMSRx, Pn_OMCR, Pn_OMCRx, (x=0-3)	The unused control fields within the same nibble pin group as available pins are able to control Pn_OUT. Otherwise, these unused control fields do not have an influence on Pn_OUT.
Pn_LPCRx	These registers are only implemented for pad pairs with LVDS functionality.
Pn_PCSR	These register are always implemented but mostly reserved. They configure Tricore/SCR control, Ethernet fast RGMII/RMII/MII mode, VADC PDD/MDD feature.

**Table 485 Register Overview - Pn (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
OUT	Port n Output Register	000 <sub>H</sub>	U,SV	U,SV,P	Application Reset	<a href="#">20</a>
OMR	Port n Output Modification Register	004 <sub>H</sub>	U,SV	U,SV,P	Application Reset	<a href="#">21</a>
ID	Port n Identification Register	008 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">7</a>
	Reserved (004 <sub>H</sub> Byte)	00C <sub>H</sub>	BE	BE		
IOCR0	Port n Input/Output Control Register 0	010 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">8</a>	<a href="#">8</a>
IOCR4	Port n Input/Output Control Register 4	014 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">10</a>	<a href="#">10</a>
IOCR8	Port n Input/Output Control Register 8	018 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">10</a>	<a href="#">10</a>

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

**Table 485 Register Overview - Pn (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
IOCR12	Port n Input/Output Control Register 12	01C <sub>H</sub>	U,SV	U,SV,P	See page 11	11
	Reserved (004 <sub>H</sub> Byte)	020 <sub>H</sub>	BE	BE		
IN	Port n Input Register	024 <sub>H</sub>	U,SV	BE	Application Reset	31
	Reserved (004 <sub>H</sub> Byte) (x=0-5)	028 <sub>H</sub> +x*4	BE	BE		
PDR0	Port n Pad Driver Mode Register 0	040 <sub>H</sub>	U,SV	SV,E,P	See page 12	12
PDR1	Port n Pad Driver Mode Register 1	044 <sub>H</sub>	U,SV	SV,E,P	See page 14	14
	Reserved (004 <sub>H</sub> Byte) (x=0-1)	048 <sub>H</sub> +x*4	BE	BE		
ESR	Port n Emergency Stop Register	050 <sub>H</sub>	U,SV	SV,E,P	Application Reset	30
	Reserved (004 <sub>H</sub> Byte) (x=0-2)	054 <sub>H</sub> +x*4	BE	BE		
PDISC	Port n Pin Function Decision Control Register	060 <sub>H</sub>	U,SV	SV,E,P	See page 18	18
PCSR	Port n Pin Controller Select Register	064 <sub>H</sub>	U,SV	SV,SE	Application Reset	19
	Reserved (004 <sub>H</sub> Byte) (x=0-1)	068 <sub>H</sub> +x*4	BE	BE		
OMSR0	Port n Output Modification Set Register 0	070 <sub>H</sub>	U,SV	U,SV,P	Application Reset	23
OMSR4	Port n Output Modification Set Register 4	074 <sub>H</sub>	U,SV	U,SV,P	Application Reset	24
OMSR8	Port n Output Modification Set Register 8	078 <sub>H</sub>	U,SV	U,SV,P	Application Reset	24
OMSR12	Port n Output Modification Set Register 12	07C <sub>H</sub>	U,SV	U,SV,P	Application Reset	25
OMCR0	Port n Output Modification Clear Register 0	080 <sub>H</sub>	U,SV	U,SV,P	Application Reset	27
OMCR4	Port n Output Modification Clear Register 4	084 <sub>H</sub>	U,SV	U,SV,P	Application Reset	27
OMCR8	Port n Output Modification Clear Register 8	088 <sub>H</sub>	U,SV	U,SV,P	Application Reset	28
OMCR12	Port n Output Modification Clear Register 12	08C <sub>H</sub>	U,SV	U,SV,P	Application Reset	29
OMSR	Port n Output Modification Set Register	090 <sub>H</sub>	U,SV	U,SV,P	Application Reset	22

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

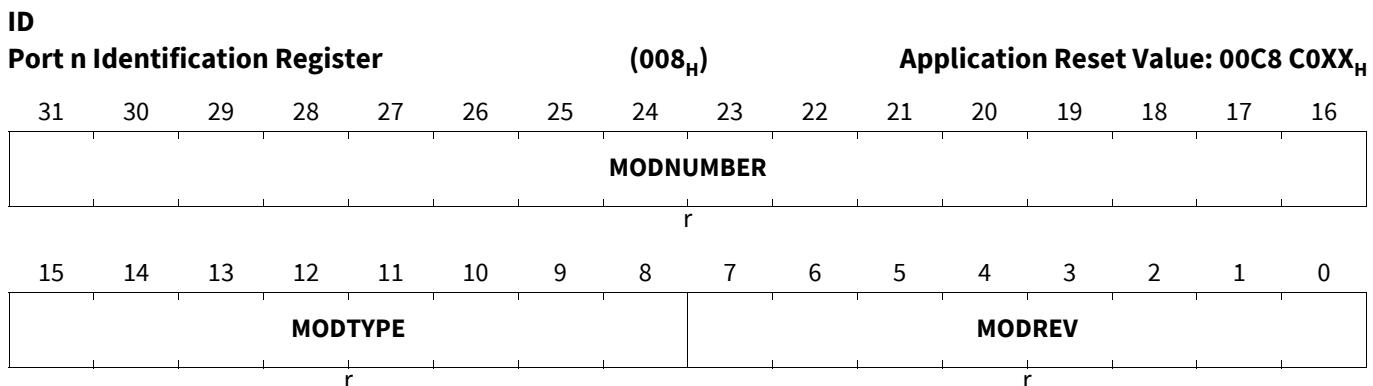
**Table 485 Register Overview - Pn (ascending Offset Address) (cont'd)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
OMCR	Port n Output Modification Clear Register	094 <sub>H</sub>	U,SV	U,SV,P	Application Reset	<b>26</b>
	Reserved (004 <sub>H</sub> Byte) (x=0-1)	098 <sub>H</sub> +x* 4	BE	BE		
LPCR <sub>x</sub>	Port n LVDS Pad Control Register x	0A0 <sub>H</sub> +x* 4	U,SV	SV,E,P	See page <b>15</b>	<b>15</b>
	Reserved (004 <sub>H</sub> Byte) (x=0-13)	0C0 <sub>H</sub> +x* 4	BE	BE		
ACCEN1	Port n Access Enable Register 1	0F8 <sub>H</sub>	U,SV	SV,SE	Application Reset	<b>32</b>
ACCEN0	Port n Access Enable Register 0	0FC <sub>H</sub>	U,SV	SV,SE	Application Reset	<b>32</b>

### 14.4.1 Module Identification Register

#### Port n Identification Register

The module Identification Register ID contains read-only information about the module version.



Field	Bits	Type	Description
<b>MODREV</b>	7:0	r	<b>Module Revision Number</b> This bit field indicates the revision number of the AURIX™ TC3xx Platform module (01 <sub>H</sub> = first revision).
<b>MODTYPE</b>	15:8	r	<b>Module Type</b> This bit field is C0 <sub>H</sub> . It defines a 32-bit module
<b>MODNUMBER</b>	31:16	r	<b>Module Number</b> This bit field defines the module identification number. The value for the Ports module is 00C8 <sub>H</sub>

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

### 14.4.2 Port Input/Output Control Registers

#### Port n Input/Output Control Register 0

The port input/output control registers select the digital output and input driver functionality and characteristics of a GPIO port pin. Port direction (input or output), pull-up, pull-down, or no pull devices for inputs, and push-pull or open-drain functionality for outputs can be selected by the corresponding bit fields PCx (x = 0-15). Each 32-bit wide port input/output control register controls four GPIO port lines:

Register Pn\_IOCRO controls the Pn.[3:0] port lines

Register Pn\_IOCRA controls the Pn.[7:4] port lines

Register Pn\_IOCRA8 controls the Pn.[11:8] port lines

Register Pn\_IOCRA12 controls the Pn.[15:12] port lines

The structure with one control bit field for each port pin located in different register bytes offers the possibility to configure the port pin functionality of a single pin with byte-oriented accesses without accessing the other PCx bit fields.

The reset values of  $1010\ 1010_H$  and  $0000\ 0000_H$  for Pn\_IOCRAx registers represents input pull-up and no input pull device (tri-state mode) being activated, respectively. The switching of the intended mode of the device is controlled by HWCFG6. When a cold reset is activated and HWCFG6=1, the port pins except P33.8, P40 and P41 are set to input pull-up mode, P33.8, P40 and P41 are in tri-state mode as long as PORST is activated. If HWCFG6=0, the pins have the default state of tri-state mode. The pad state can also be configured by software through PMSWCR5.TRISTREQ bit. In the event of a warm reset or wake-up from standby mode, PMSWCR5.TRISTREQ is not affected by reset, hence Pn\_IOCRAx registers have the reset values configured as per the last state of the TRISTREQ bit.

**Note:** *In LVDS (RX and TX) operation the IOCRA register of both pins of the LVDS pair must be configured as output, i.e.  $1xxxx_B$ . This ensures that the pull devices are disconnected and don't interfere with LVDS operation.*

Register Pn\_IOCRO controls the Pn.[3:0] port lines

#### IOCRO

#### Port n Input/Output Control Register 0 (010<sub>H</sub>) Reset Value: [Table 487](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				<b>PC3</b>			<b>0</b>			<b>PC2</b>			<b>0</b>		
				rw			r			rw			r		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				<b>PC1</b>			<b>0</b>			<b>PC0</b>			<b>0</b>		
				rw			r			rw			r		

Field	Bits	Type	Description
<b>PCx (x=0-3)</b>	$8^*x+7:8^*x+3$	rw	<b>Port Control for Pin x</b> This bit field defines the Port n line x functionality according to <a href="#">Table 488</a> .
<b>0</b>	26:24, 18:16, 10:8, 2:0	r	<b>Reserved</b> Read as 0; should be written with 0.

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

**Table 486 Access Mode Restrictions of **IOCR0** sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	PCx (x=0-3)	write access for enabled masters
Otherwise (default)	r	PCx (x=0-3)	

**Table 487 Reset Values of **IOCR0****

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	HWCFG6 is 0 (tri-state mode)
Application Reset	1010 1010 <sub>H</sub>	HWCFG6 is 1 (input pull-up mode)

### Port Control Coding

**Table 488** describes the coding of the PCx bit fields that determine the port line functionality.

**Table 488 PCx Coding**

PCx[4:0]	I/O	Characteristics	Selected Pull-up / Pull-down / Selected Output Function
0XX00 <sub>B</sub>	Input	-	No input pull device connected, tri-state mode
0XX01 <sub>B</sub>			Input pull-down device connected
0XX10 <sub>B</sub>			Input pull-up device connected <sup>1)</sup>
0XX11 <sub>B</sub>			No input pull device connected, tri-state mode
10000 <sub>B</sub>	Output	Push-pull	General-purpose output
10001 <sub>B</sub>			Alternate output function 1
10010 <sub>B</sub>			Alternate output function 2
10011 <sub>B</sub>			Alternate output function 3
10100 <sub>B</sub>			Alternate output function 4
10101 <sub>B</sub>			Alternate output function 5
10110 <sub>B</sub>			Alternate output function 6
10111 <sub>B</sub>			Alternate output function 7
11000 <sub>B</sub>	Open-drain	Open-drain	General-purpose output
11001 <sub>B</sub>			Alternate output function 1
11010 <sub>B</sub>			Alternate output function 2
11011 <sub>B</sub>			Alternate output function 3
11100 <sub>B</sub>			Alternate output function 4
11101 <sub>B</sub>			Alternate output function 5
11110 <sub>B</sub>			Alternate output function 6
11111 <sub>B</sub>			Alternate output function 7

1) This is the default pull device setting after reset for powertrain applications.

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

### Port n Input/Output Control Register 4

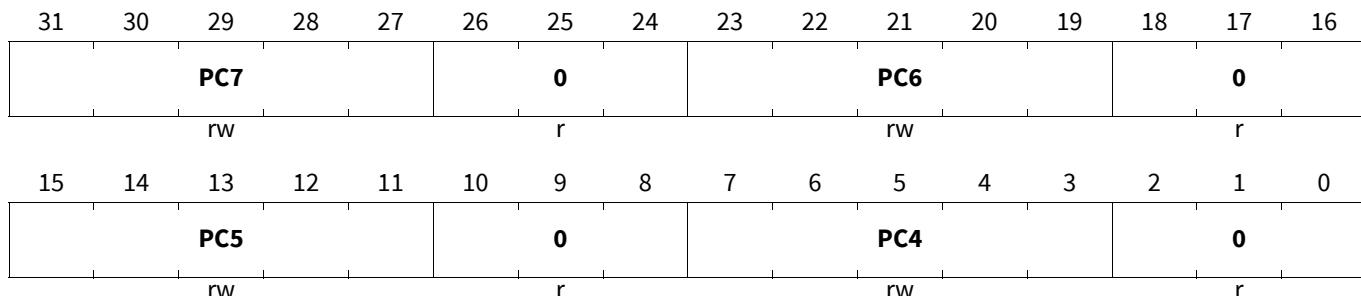
Register Pn\_IOCR4 controls the Pn.[7:4] port lines

#### IOCR4

##### Port n Input/Output Control Register 4

(014<sub>H</sub>)

Reset Value: [Table 490](#)



Field	Bits	Type	Description
PCx (x=4-7)	8*x-25:8*x-29	rw	<b>Port Control for Port n Pin x</b> This bit field defines the Port n line x functionality according to <a href="#">Table 488</a> .
0	26:24, 18:16, 10:8, 2:0	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 489 Access Mode Restrictions of IOCR4 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	PCx (x=4-7)	write access for enabled masters
Otherwise (default)	r	PCx (x=4-7)	

**Table 490 Reset Values of IOCR4**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	HWCFG6 is 0 (tri-state mode)
Application Reset	1010 1010 <sub>H</sub>	HWCFG6 is 1 (input pull-up mode)

### Port n Input/Output Control Register 8

Register Pn\_IOCR8 controls the Pn.[11:8] port lines

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

### IOCR8

#### Port n Input/Output Control Register 8

(018<sub>H</sub>)Reset Value: [Table 492](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
PC11						0	PC10						0			
rw				r				rw				r				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
PC9						0	PC8						0			
rw				r				rw				r				

Field	Bits	Type	Description
PCx (x=8-11)	8*x-57:8*x-61	rw	<b>Port Control for Port n Pin x</b> This bit field defines the Port n line x functionality according to <a href="#">Table 488</a> .
0	26:24, 18:16, 10:8, 2:0	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 491 Access Mode Restrictions of IOCR8 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw PCx (x=8-11)		write access for enabled masters
Otherwise (default)	r PCx (x=8-11)		

**Table 492 Reset Values of IOCR8**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	HWCFG6 is 0 (tri-state mode)
Application Reset	1010 1010 <sub>H</sub>	HWCFG6 is 1 (input pull-up mode)

### Port n Input/Output Control Register 12

Register Pn\_IOCR12 controls the Pn.[15:12] port lines

### IOCR12

#### Port n Input/Output Control Register 12

(01C<sub>H</sub>)Reset Value: [Table 494](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
PC15						0	PC14						0			
rw				r				rw				r				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
PC13						0	PC12						0			
rw				r				rw				r				

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
<b>PCx (x=12-15)</b>	8*x-89:8*x-93	rw	<b>Port Control for Port n Pin x</b> This bit field defines the Port n line x functionality according to <a href="#">Table 488</a> .
<b>0</b>	26:24, 18:16, 10:8, 2:0	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 493 Access Mode Restrictions of [IOCR12](#) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	PCx (x=12-15)	write access for enabled masters
Otherwise (default)	r	PCx (x=12-15)	

**Table 494 Reset Values of [IOCR12](#)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	HWCFG6 is 0 (tri-state mode)
Application Reset	1010 1010 <sub>H</sub>	HWCFG6 is 1 (input pull-up mode)

### 14.4.3 Pad Driver Mode Register

#### Port n Pad Driver Mode Register 0

**PDR0**

**Port n Pad Driver Mode Register 0**

(040<sub>H</sub>)

Reset Value: [Table 496](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>PL7</b>	<b>PD7</b>	<b>PL6</b>	<b>PD6</b>	<b>PL5</b>	<b>PD5</b>	<b>PL4</b>	<b>PD4</b>								
rw	rw	rw	rw	rw	rw	rw	rw	rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>PL3</b>	<b>PD3</b>	<b>PL2</b>	<b>PD2</b>	<b>PL1</b>	<b>PD1</b>	<b>PL0</b>	<b>PD0</b>								
rw	rw	rw	rw	rw	rw	rw	rw	rw							

Field	Bits	Type	Description
<b>PDx (x=0-7)</b>	4*x+1:4*x	rw	<b>Pad Driver Mode for Pin x</b>
<b>PLx (x=0-7)</b>	4*x+3:4*x+2	rw	<b>Pad Level Selection for Pin x</b>

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

**Table 495 Access Mode Restrictions of PDR0 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN and Supervisor Mode and ENDINIT	rw	PDX (x=0-7), PLx (x=0-7)	write access for enabled masters
Otherwise (default)	r	PDX (x=0-7), PLx (x=0-7)	

**Table 496 Reset Values of PDR0**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
After SSW execution	---- ---- <sub>H</sub>	Initial value package dependent

### Output Characteristics

The pad structure of the GPIO lines offers the possibility to select the output driver strength and the slew rate. These two parameters are controlled by the PDX bit fields in the pad driver mode registers Pn\_PDR0/1 for output modes. The available modes depend on the respective pad type.

**Table 497 Pad Driver Mode Selection for RFast Pads**

PDX.1	PDX.0	Speed Grade	Driver Setting
0	0	1	Strong driver, sharp edge ("ss")
0	1	2	Strong driver, medium edge ("sm")
1	0	3	Medium driver ("m")
1	1	4	RGMII driver.

**Table 498 Pad Driver Mode Selection for Fast Pads**

PDX.1	PDX.0	Speed Grade	Driver Setting
0	0	1	Strong driver, sharp edge ("ss")
0	1	2	Strong driver, medium edge ("sm")
1	0	3	Medium driver ("m")
1	1	4	TC39x A-Step: Medium driver ("m") Else: Reserved when operating as output. When operating as input see below "Pad Level Selection for Input Function".

**Table 499 Pad Driver Mode Selection for Slow Pads**

PDX.1	PDX.0	Speed Grade	Driver Setting
X	0	1	Medium driver, sharp edge ("sm") <sup>1)</sup>
X	1	2	Medium driver ("m")

1) This setting is marked "sm" as the electrical characteristics are identical to the strong driver medium edge setting. The Data Sheet contains also only common "sm" tables.

**Note:** The Data Sheet describes the DC characteristics of all pad classes.

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

### TTL/Automotive Input Selection

The input function can operate with different VIH and VIL levels depending on the pad supply voltage, the pad type and the selection done by the PLx bits of the Pn\_PDRx as of [Table 500](#). PLx.1 changes additionally the pull-up and pull-down resistors.

**Table 500 Pad Level Selection for Input Function**

PLx.1	PLx.0	Input Levels
0	X	Automotive level “AL”.
1	0	TTL level for 5V pad supply. Degraded TTL level used for CIF when pad supply is 3.3V
1	1	TTL level for 3.3V pad supply.
X	X	Only for pads with RGMII input buffer (marked “RGMII_Input” in the pinning table): <ul style="list-style-type: none"> <li>when PDx.1=1 and PDx.0=1 the input level RGMII is selected.</li> <li>for other PDx values the input level is determined by PLx as for all other pads (first three rows of this table).</li> </ul>

### LVDS

The default CMOS mode can be switched to LVDS mode in LVDS pads through the LPCRx register.

### Pad Driver Mode Registers

This is the general description of the PDR registers. Each port contains its own specific PDR registers, described additionally at each port, that can contain between one and eight PDx fields for PDR0 and PDR1 registers, respectively. Each PDx field controls 1 pin. For coding of PDx, see [Table 497](#), [Table 498](#) and [Table 499](#). Similarly, each PLx bit controls 1 pin. For coding of PLx, see [Table 500](#).

The boot software configures the reset value of Pn\_PDR0 and Pn\_PDR1 registers from 0000 0000<sub>H</sub> to 2222 2222<sub>H</sub> except for analog ports and if the package doesn't make any of the related pins available. The resulting value depends on the implemented port width. The documented value is valid for the largest package.

### Port n Pad Driver Mode Register 1

#### PDR1

#### Port n Pad Driver Mode Register 1

(044<sub>H</sub>)

Reset Value: [Table 502](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>PL15</b>	<b>PD15</b>	<b>PL14</b>	<b>PD14</b>	<b>PL13</b>	<b>PD13</b>	<b>PL12</b>	<b>PD12</b>								
rw	rw	rw	rw	rw	rw	rw	rw	rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>PL11</b>	<b>PD11</b>	<b>PL10</b>	<b>PD10</b>	<b>PL9</b>	<b>PD9</b>	<b>PL8</b>	<b>PD8</b>								
rw	rw	rw	rw	rw	rw	rw	rw	rw							

Field	Bits	Type	Description
<b>PDx (x=8-15)</b>	4*x-31:4*x-32	rw	<b>Pad Driver Mode for Pin x</b>

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PLx (x=8-15)	4*x-29:4*x-30	rw	<b>Pad Level Selection for Pin x</b>

**Table 501** Access Mode Restrictions of PDR1 sorted by descending priority

<b>Mode Name</b>	<b>Access Mode</b>		<b>Description</b>
Master enabled in ACCEN and Supervisor Mode and ENDINIT	rw	PDx (x=8-15), PLx (x=8-15)	write access for enabled masters
Otherwise (default)	r	PDx (x=8-15), PLx (x=8-15)	

**Table 502 Reset Values of PDR1**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
Application Reset	0000 0000 <sub>H</sub>	
After SSW execution	----- <sub>H</sub>	Initial value package dependent

#### **14.4.4 LVDS Pad Control Register**

## **Port n LVDS Pad Control Register x**

The LVDS Pad Control Register controls the RX or TX functions of the LVDS pads. For usage of RX pad, bit field [7:0] are applicable. If used for TX pad, bit field [15:7] apply.

The sleep functionality of the LVDS pads is not controllable via this register. This is exclusively controlled by the HSCT module when this is connected.

The register  $x$  controls in general the pad pair  $2^*x$  and  $2^*x+1$  of the port  $n$ .

Exceptionally when available the pad pair P14.9 and P14.10 is controlled by P14\_LPCR5.

**Attention:** The bit field P21\_LPCR2.PS configures the pad supply for the LVDS bias distributor for all (not-RIF) LVDS pads and for the oscillator. Therefore even if no LVDS pad is used, this field has to be configured to the correct pad supply level.

LPCR<sub>x</sub> (x=0-7)

## **Port n LVDS Pad Control Register x**

$$(0A0_H + x^*4)$$

## **Reset Value: Table 504**

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
REN_CTRL	0	rw	<b>LVDS RX_EN controller</b> The LVDS RX_EN control function can be selected from the Port (default) or HSCT module (where this is connected). 0 <sub>B</sub> Port controlled 1 <sub>B</sub> HSCT controlled (reserved where no HSCT is connected)
RX_EN	1	rw	<b>Enable Receive LVDS</b> Enable the receive LVDS / disable CMOS path. If this bit is set to 0 – no transfer from the LVDS sender can be received and the receiver LVDS is in low power state. 0 <sub>B</sub> disable LVDS / enable CMOS mode (reserved for pads without CMOS input stage) 1 <sub>B</sub> enable LVDS / disable CMOS mode
TERM	2	rw	<b>Select Receiver Termination Mode</b> Selects a suitable internal load resistor between both pads. 0 <sub>B</sub> external termination - on the PCB 1 <sub>B</sub> 100 Ω Receiver internal termination
LRXTERM	5:3	rw	<b>LVDS RX Poly-resistor configuration value</b> Programming bits for the on die poly resistor termination. The value is configured during production test. Each chip configuration on this bit field is unique and configured during production testing.  <i>Note:</i> The configuration value shall not be changed by user after start-up for a guaranteed behavior.
LVDSM	6	rw	<b>LVDS-M Mode</b> Selects reduced frequency mode “LVDS-M” of the receiver. This mode reduces the static current of the RX pad. The max data rate is reduced to 160 Mbps (80 MHz). 0 <sub>B</sub> LVDS-H Mode 1 <sub>B</sub> LVDS-M Mode
PS	7	rw	<b>Pad Supply Selection</b> Selects between 5V or 3.3V supply on $V_{EXT}$ for the pad-pair. Used in RX and TX pads! 0 <sub>B</sub> 3.3V supply 1 <sub>B</sub> 5V supply
TEN_CTRL	8	rw	<b>LVDS TX_EN controller</b> The LVDS TX_EN control function can be selected from the Port (default) or HSCT module (where this is connected). 0 <sub>B</sub> Port controlled 1 <sub>B</sub> HSCT controlled (reserved where no HSCT is connected)
TX_EN	9	rw	<b>Enable Transmit LVDS</b> Enable the transmit LVDS / disable CMOS path. If this bit is set to 0 - no transfer on LVDS data path can be initiated and the LVDS driver is disabled (powered down). 0 <sub>B</sub> disable LVDS / enable CMOS mode 1 <sub>B</sub> enable LVDS / disable CMOS mode

### General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
<b>VDIFFADJ</b>	11:10	rw	<b>LVDS Output Amplitude Tuning</b> With these two configuration bits the LVDS output current/amplitude can be adjusted. The voltage swing depending on VDIFFADJ setting is documented in the Data Sheet, see parameter $V_{OD}$ .
<b>VOSDYN</b>	12	rw	<b>Tune Bit of VOS Control Loop Static/Dynamic</b> Tune bit to change $V_{OS}$ control loop between static and dynamic mode. Don't change reset value.
<b>VOSEXT</b>	13	rw	<b>Tune Bit of VOS Control Loop Internal/External</b> Tune bit to change $V_{OS}$ control loop. Don't change reset value.
<b>TX_PD</b>	14	rw	<b>LVDS Power Down</b> Unused in this device. LVDS disabled by TX_EN means power down. $0_B$ LVDS power on $1_B$ LVDS power down (default)
<b>TX_PWDPD</b>	15	rw	<b>Enable TX Power down pull down.</b> This function disables or enables the LVDS pull down resistor. The application code must disable TX power down pull down resistor with a power up. With a LVDS Power Down configuration the pull down function must be enabled, if required. $0_B$ disabled TX Power down pull down resistor. $1_B$ enabled TX Power down pull down resistor.
<b>0</b>	31:16	r	<b>Reserved</b> Read as 0; should be written with 0

**Table 503 Access Mode Restrictions of  $LPCRx$  ( $x=0-7$ ) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN and Supervisor Mode and ENDINIT	rw	LRXTERM, LVDSM, PS, REN_CTRL, RX_EN, TEN_CTRL, TERM, TX_EN, TX_PD, TX_PWDPD, VDIFFADJ, VOSDYN, VOSEXT	write access for enabled masters
Otherwise (default)	r	LRXTERM, LVDSM, PS, REN_CTRL, RX_EN, TEN_CTRL, TERM, TX_EN, TX_PD, TX_PWDPD, VDIFFADJ, VOSDYN, VOSEXT	

**Table 504 Reset Values of  $LPCRx$  ( $x=0-7$ )**

Reset Type	Reset Value	Note
Application Reset	0000 54C0 <sub>H</sub>	
After SSW execution	0000 ---- <sub>H</sub>	Initial value depends on RX/TX and trimming

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

### 14.4.5 Pin Function Decision Control Register

#### Port n Pin Function Decision Control Register

The pad structure of the GPIO lines offers the possibility to disable/enable port pad, select digital input or analog ADC input functionalities. Note that Class S pads have different characteristics than other digital input pads. For analog inputs, setting PDISx to 1 disables the Schmitt trigger input buffer, which would otherwise reduce analog input accuracy. For the ADC diagnostic features “PDD” and “MD” however the corresponding PDISx needs to be 0 to allow activation of their pull resistors. This feature can be controlled by individual bits in the Pn\_PDISC register, independently from input/output and pull-up/pull-down control functionality as programmed in the Pn\_IOCR register. One Pn\_PDISC register is assigned to each port.

**Note:** After reset, all Px\_PDISC registers have the reset value of  $0000\ 0000_H$ . The startup software enables only the pads with digital input/output functionality which are available in that package. P40\_PDISC and P41\_PDISC are configured by the SSW for analog input function (kept disabled). The documented reset value shows the value in the largest package.

#### PDISC

##### Port n Pin Function Decision Control Register (060<sub>H</sub>)

Reset Value: [Table 506](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
								r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>PDIS15</b>	<b>PDIS14</b>	<b>PDIS13</b>	<b>PDIS12</b>	<b>PDIS11</b>	<b>PDIS10</b>	<b>PDIS9</b>	<b>PDIS8</b>	<b>PDIS7</b>	<b>PDIS6</b>	<b>PDIS5</b>	<b>PDIS4</b>	<b>PDIS3</b>	<b>PDIS2</b>	<b>PDIS1</b>	<b>PDIS0</b>
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
<b>PDISx (x=0-15)</b>	x	rw	<b>Pin Function Decision Control for Pin x</b> This bit selects the function of the port pad. 0 <sub>B</sub> Digital functionality of pad Pn.x is enabled. 1 <sub>B</sub> Digital functionality (including pull resistors) of pad Pn.x is disabled. Analog input function (where this is available) can be used.
<b>0</b>	31:16	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 505 Access Mode Restrictions of PDISC sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN and Supervisor Mode and ENDINIT	rw	PDISx (x=0-15)	write access for enabled masters
Otherwise (default)	r	PDISx (x=0-15)	

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

**Table 506 Reset Values of PDISC**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
After SSW execution	0000 ----- <sub>H</sub>	Initial value package dependent

### 14.4.6 Pin Controller Select Register

#### Port n Pin Controller Select Register

This register has different functionality in each port:

- In Ports shared with the standby controller (SCR) it selects if the SCR or the Tricore system control data and control functions of these port lines.
- In Ports with analog inputs to the EVADC it enables control of pull by the EVADC for the Pull Down Diagnostics (PDD) / Multiplexer Diagnostics (MD) feature.
- In Ports with Ethernet output it selects between alternate output and fast RGMII/RMII/MII mode.
- In Ports with SMU FSP pin (P33.8) the PCSR.SEL bit enables the SMU to override pad configuration signals. Therefore this bit has the reset value 1<sub>B</sub> and shall be kept 1<sub>B</sub> by the application. The SMU override is documented in the SMU chapter (see SMU\_PCTL.PCFG and Figure “SMU/PAD Control Interface to the PADs”).

#### PCSR

**Port n Pin Controller Select Register (064<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LCK</b>								<b>0</b>							
rw								r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>SEL15</b>	<b>SEL14</b>	<b>SEL13</b>	<b>SEL12</b>	<b>SEL11</b>	<b>SEL10</b>	<b>SEL9</b>	<b>SEL8</b>	<b>SEL7</b>	<b>SEL6</b>	<b>SEL5</b>	<b>SEL4</b>	<b>SEL3</b>	<b>SEL2</b>	<b>SEL1</b>	<b>SEL0</b>
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
<b>SELx (x=0-15)</b>	x	rw	<p><b>Output Select for Pin x</b></p> <p>Depending on the port this bit enables or disables Tricore/SCR control, the EVADC pull control for Pull Down Diagnostics (PDD) / Multiplexer Diagnostics (MD) feature, SMU override or alternate/fast Ethernet output.</p> <p>0<sub>B</sub> Tricore selected for data and control of pin x and not SCR. Disable VADC PDD/MD feature of pin x. Ethernet output via ports alternate output of pin x. Disable SMU override of pad configuration for FSP pin x.</p> <p>1<sub>B</sub> SCR selected for data and control of pin x (which can with its register SPAREINOUT0.0 also enable VADC PDD / MD feature of pin x). Enable VADC PDD/MD feature of pin x. Ethernet output via fast RGMII/RMII/MII mode of pin x. Enable SMU to override pad configuration for FSP pin x.</p>

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
LCK	31	rh	<p><b>Lock Status</b></p> <p>This bit indicates if the register can be updated with a new value or if the register is locked due to an ongoing transfer to the SCR and a write action from the bus has no effect.</p> <p>In Ports without SCR overlay this bit is always <math>0_B</math>.</p> <p><math>0_B</math> The register is unlocked and can be updated.</p> <p><math>1_B</math> The register is locked (a write transfer to SCR is ongoing) and can not be updated.</p>
0	30:16	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 507 Access Mode Restrictions of PCSR sorted by descending priority**

Mode Name	Access Mode		Description
Supervisor Mode and Safety ENDINIT	rh	LCK	write access only for masters with supervisor mode
	rw	SELx (x=0-15)	
Otherwise (default)	r	SELx (x=0-15)	
	rh	LCK	

### 14.4.7 Port Output Register

#### Port n Output Register

The port output register determines the value of a GPIO pin when it is selected by Pn\_IOCRx as output. Writing a 0 to a Pn\_OUT.Px (x = 0-15) bit position delivers a low level at the corresponding output pin. A high level is output when the corresponding bit is written with a 1. Note that the bits of Pn\_OUT.Px can be individually set or cleared by writing appropriate values into the port output modification set register Pn\_OMSR or port output modification clear register Pn\_OMCR, respectively. The Pn\_OUT.Px bits can also be set, cleared or toggled with register Pn\_OMR within the same write operation.

#### OUT

Port n Output Register																(000 <sub>H</sub> )	Application Reset Value: 0000 0000 <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	0																
																r																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0	
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh																	

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
Px (x=0-15)	x	rwh	<b>Output Bit x</b> This bit determines the level at the output pin Pn.x if the output is selected as GPIO output. Pn.x can also be set or cleared by control bits of the Pn_OMSR, Pn_OMCR or Pn_OMR registers. 0 <sub>B</sub> The output level of Pn.x is 0. 1 <sub>B</sub> The output level of Pn.x is 1.
0	31:16	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 508 Access Mode Restrictions of OUT sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rwh	Px (x=0-15)	write access for enabled masters
Otherwise (default)	rh	Px (x=0-15)	

### 14.4.8 Port Output Modification Register

#### Port n Output Modification Register

The port output modification register contains control bits that make it possible to individually set, clear or toggle the logic state of a single port line by manipulating the output register.

#### OMR

Port n Output Modification Register (004 <sub>H</sub> )																Application Reset Value: 0000 0000 <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	PCL15	PCL14	PCL13	PCL12	PCL11	PCL10	PCL9	PCL8	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	PS15	PS14	PS13	PS12	PS11	PS10	PS9	PS8	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0		

Field	Bits	Type	Description
PSx (x=0-15)	x	w0	<b>Set Bit x</b> Setting this bit will set or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. The function of this bit is shown in <b>Table 510</b> . 0 <sub>B</sub> No operation 1 <sub>B</sub> Sets or toggles Pn_OUT.Px.

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PCLx (x=0-15)	x+16	w0	<p><b>Clear Bit x</b></p> <p>Setting this bit will clear or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. The function of this bit is shown in <a href="#">Table 510</a>.</p> <p>0<sub>B</sub> No operation 1<sub>B</sub> Clears or toggles Pn_OUT.Px.</p>

**Table 509 Access Mode Restrictions of OMR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	w0	PCLx (x=0-15), PSx (x=0-15)	write access for enabled masters
Otherwise (default)	r0	PCLx (x=0-15), PSx (x=0-15)	

Note: Register Pn\_OMR is virtual and does not contain any flip-flop. A read action delivers the value of 0. One 8 or 16-bits write behaves as a 32-bit write padded with zeros.

**Table 510 Function of the Bits PCLx and PSx**

PCLx	PSx	Function
0	0	Bit Pn_OUT.Px is not changed.
0	1	Bit Pn_OUT.Px is set.
1	0	Bit Pn_OUT.Px is reset.
1	1	Bit Pn_OUT.Px is toggled.

### 14.4.9 Port Output Modification Set Register

#### Port n Output Modification Set Register

The port output modification set register contains control bits that make it possible to individually set the logic state of a single port line by manipulating the output register.

Note: Register Pn\_OMSR is virtual and does not contain any flip-flop. A read action delivers the value of 0. One 8 or 16-bits write behaves as a 32-bit write padded with zeros.

#### OMSR

**Port n Output Modification Set Register (090<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS15	PS14	PS13	PS12	PS11	PS10	PS9	PS8	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PSx (x=0-15)	x	w0	<b>Set Bit x</b> Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. $0_B$ No operation $1_B$ Sets Pn_OUT.Px
0	31:16	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 511 Access Mode Restrictions of OMSR sorted by descending priority**

Mode Name	Access Mode	Description
Master enabled in ACCEN	w0	PSx (x=0-15) write access for enabled masters
Otherwise (default)	r0	PSx (x=0-15)

### 14.4.10 Port Output Modification Set Registers

#### Port n Output Modification Set Register 0

The port output modification set register x, (x = 0, 4, 8, 12) contains control bits to individually set the logic state of a single port line by manipulating the output register.

**Note:** Registers Pn\_OMSRx (x = 0, 4, 8, 12) are virtual and does not contain any flip-flop. A read action delivers the value of 0. One 8 or 16-bits write behaves as a 32-bit write padded with zeros.

Register Pn\_OMSR0 sets the logic state of Pn.[3:0] port lines

#### OMSR0

Port n Output Modification Set Register 0 (070 <sub>H</sub> )																Application Reset Value: 0000 0000 <sub>H</sub>			
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16																0000 0000 <sub>H</sub>			
0																0000 0000 <sub>H</sub>			

Field	Bits	Type	Description
PSx (x=0-3)	x	w0	<b>Set Bit x</b> Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. $0_B$ No operation $1_B$ Sets Pn_OUT.Px
0	31:4	r	<b>Reserved</b> Read as 0; should be written with 0.

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

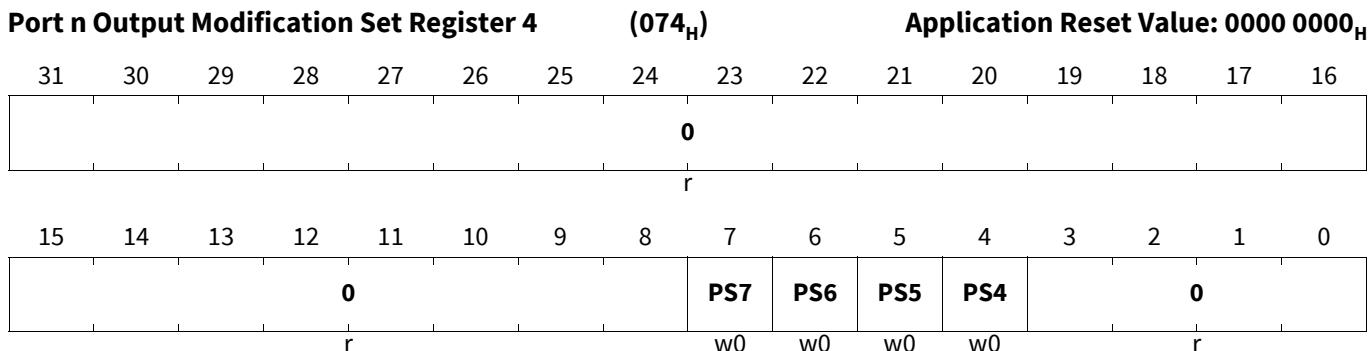
Table 512 Access Mode Restrictions of OMSR0 sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	w0	PSx (x=0-3)	write access for enabled masters
Otherwise (default)	r0	PSx (x=0-3)	

## Port n Output Modification Set Register 4

Register Pn\_OMSR4 sets the logic state of Pn.[7:4] port lines

## OMSR4



Field	Bits	Type	Description
PSx (x=4-7)	x	w0	<b>Set Bit x</b> Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0 <sub>B</sub> No operation 1 <sub>B</sub> Sets Pn_OUT.Px
0	3:0, 31:8	r	<b>Reserved</b> Read as 0; should be written with 0.

Table 513 Access Mode Restrictions of OMSR4 sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	w0	PSx (x=4-7)	write access for enabled masters
Otherwise (default)	r0	PSx (x=4-7)	

## Port n Output Modification Set Register 8

Register Pn\_OMSR8 sets the logic state of Pn.[11:8] port lines

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

### OMSR8

#### Port n Output Modification Set Register 8 (078<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
r		0		PS11	PS10	PS9	PS8					0	r		

Field	Bits	Type	Description
PSx (x=8-11)	x	w0	<b>Set Bit x</b> Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0 <sub>B</sub> No operation 1 <sub>B</sub> Sets Pn_OUT.Px
0	7:0, 31:12	r	<b>Reserved</b> Read as 0; should be written with 0.

Table 514 Access Mode Restrictions of OMSR8 sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	w0	PSx (x=8-11)	write access for enabled masters
Otherwise (default)	r0	PSx (x=8-11)	

### Port n Output Modification Set Register 12

Register Pn\_OMSR12 sets the logic state of Pn.[15:12] port lines

### OMSR12

#### Port n Output Modification Set Register 12 (07C<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS15	PS14	PS13	PS12						0	r					

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PSx (x=12-15)	x	w0	<b>Set Bit x</b> Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0 <sub>B</sub> No operation 1 <sub>B</sub> Sets Pn_OUT.Px
0	11:0, 31:16	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 515 Access Mode Restrictions of OMSR12 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	w0	PSx (x=12-15)	
Otherwise (default)	r0	PSx (x=12-15)	

### 14.4.11 Port Output Modification Clear Register

#### Port n Output Modification Clear Register

The port output modification clear register contains control bits that make it possible to individually clear the logic state of a single port line by manipulating the output register.

**Note:** Register Pn\_OMCR is virtual and does not contain any flip-flop. A read action delivers the value of 0. One 8 or 16-bits write behaves as a 32-bit write padded with zeros.

#### OMCR

<b>Port n Output Modification Clear Register (094<sub>H</sub>)</b>																<b>Application Reset Value: 0000 0000<sub>H</sub></b>			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16				
PCL15	PCL14	PCL13	PCL12	PCL11	PCL10	PCL9	PCL8	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0				
w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0	w0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
								0											
									r										

Field	Bits	Type	Description
PCLx (x=0-15)	x+16	w0	<b>Clear Bit x</b> Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0 <sub>B</sub> No operation 1 <sub>B</sub> Clears Pn_OUT.Px.
0	15:0	r	<b>Reserved</b> Read as 0; should be written with 0

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

**Table 516 Access Mode Restrictions of OMCR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	w0	PCLx (x=0-15)	write access for enabled masters
Otherwise (default)	r0	PCLx (x=0-15)	

### 14.4.12 Port Output Modification Clear Registers

#### Port n Output Modification Clear Register 0

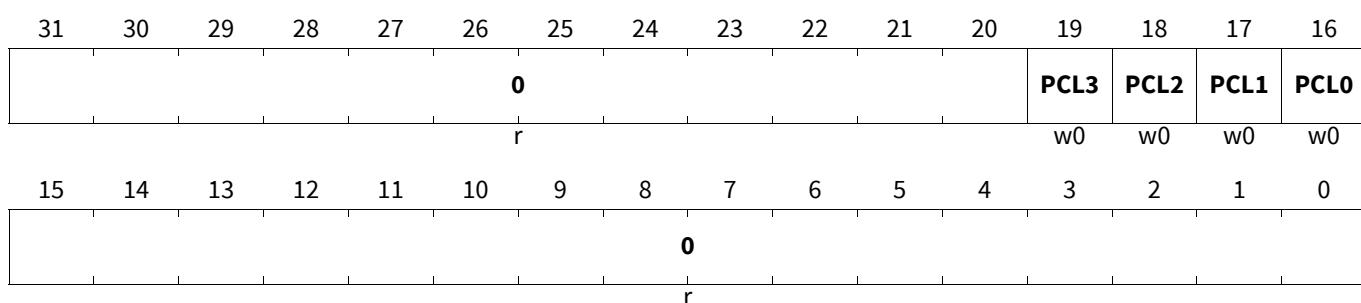
The port output modification clear register x, (x = 0, 4, 8, 12) contains control bits to individually clear the logic state of a single port line by manipulating the output register.

**Note:** Registers  $Pn\_OMCRx$  ( $x = 0, 4, 8, 12$ ) are virtual and does not contain any flip-flop. A read action delivers the value of 0. One 8 or 16-bits write behaves as a 32-bit write padded with zeros.

Register  $Pn\_OMCR0$  clears the logic state of  $Pn.[3:0]$  port lines

#### OMCRO

#### Port n Output Modification Clear Register 0 (080<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
PCLx (x=0-3)	x+16	w0	<b>Clear Bit x</b> Setting this bit will clear the corresponding bit in the port output register $Pn\_OUT$ . Read as 0. $0_B$ No operation $1_B$ Clears $Pn\_OUT.Px$
0	15:0, 31:20	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 517 Access Mode Restrictions of OMCRO sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	w0	PCLx (x=0-3)	write access for enabled masters
Otherwise (default)	r0	PCLx (x=0-3)	

#### Port n Output Modification Clear Register 4

Register  $Pn\_OMCR4$  clears the logic state of  $Pn.[7:4]$  port lines

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

### OMCR4

#### Port n Output Modification Clear Register 4 (084<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								<b>PCL7</b>	<b>PCL6</b>	<b>PCL5</b>	<b>PCL4</b>			<b>0</b>	
r							w0	w0	w0	w0	w0		r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							<b>0</b>								
r															

Field	Bits	Type	Description
<b>PCLx (x=4-7)</b>	x+16	w0	<b>Clear Bit x</b> Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. $0_B$ No operation $1_B$ Clears Pn_OUT.Px
<b>0</b>	19:0, 31:24	r	<b>Reserved</b> Read as 0; should be written with 0

Table 518 Access Mode Restrictions of OMCR4 sorted by descending priority

Mode Name	Access Mode			Description
Master enabled in ACCEN	w0	PCLx (x=4-7)		write access for enabled masters
Otherwise (default)	r0	PCLx (x=4-7)		

### Port n Output Modification Clear Register 8

Register Pn\_OMCR8 clears the logic state of Pn.[11:8] port lines

### OMCR8

#### Port n Output Modification Clear Register 8 (088<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				<b>PCL11</b>	<b>PCL10</b>	<b>PCL9</b>	<b>PCL8</b>					<b>0</b>			
r				w0	w0	w0	w0					r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							<b>0</b>								
r															

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PCLx (x=8-11)	x+16	w0	<b>Clear Bit x</b> Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0 <sub>B</sub> No operation 1 <sub>B</sub> Clears Pn_OUT.Px
0	23:0, 31:28	r	<b>Reserved</b> Read as 0; should be written with 0

**Table 519 Access Mode Restrictions of OMCR8 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	w0	PCLx (x=8-11)	write access for enabled masters
Otherwise (default)	r0	PCLx (x=8-11)	

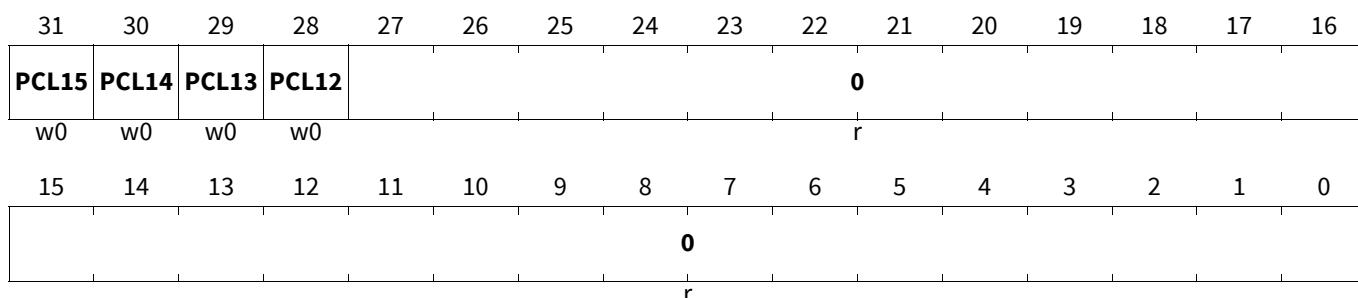
### Port n Output Modification Clear Register 12

Register Pn\_OMCR12 clears the logic state of Pn.[15:12] port lines

#### OMCR12

#### Port n Output Modification Clear Register 12 (08C<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
PCLx (x=12-15)	x+16	w0	<b>Clear Bit x</b> Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0 <sub>B</sub> No operation 1 <sub>B</sub> Clears Pn_OUT.Px
0	27:0	r	<b>Reserved</b> Read as 0; should be written with 0

**Table 520 Access Mode Restrictions of OMCR12 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	w0	PCLx (x=12-15)	write access for enabled masters
Otherwise (default)	r0	PCLx (x=12-15)	

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

### 14.4.13 Emergency Stop Register

#### Port n Emergency Stop Register

**ESR**

Port n Emergency Stop Register																(050 <sub>H</sub> )	Application Reset Value: 0000 0000 <sub>H</sub>																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	0	r	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

Field	Bits	Type	Description
ENx (x=0-15)	x	rw	<b>Emergency Stop Enable for Pin x</b> This bit enables the emergency stop function for all GPIO lines. If the emergency stop condition is met and enabled, the output selection is automatically switched from alternate output function to GPIO input function. 0 <sub>B</sub> Emergency stop function for Pn.x is disabled. 1 <sub>B</sub> Emergency stop function for Pn.x is enabled.
0	31:16	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 521 Access Mode Restrictions of ESR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN and Supervisor Mode and ENDINIT	rw	ENx (x=0-15)	write access for enabled masters
Otherwise (default)	r	ENx (x=0-15)	

Most GPIO lines have an emergency stop logic implemented (see Figure “General Structure of a Port Pin” in the Family chapter).

Each of these GPIO lines has its own emergency stop enable bit ENx that is located in the emergency stop register Pn\_ESR of Port n. If the emergency stop signal becomes active, one of two states can be selected:

- Emergency stop function disabled (ENx = 0):  
The output line remains connected (alternate function).
- Emergency stop function enabled (ENx = 1):  
The mapped output function is disconnected and the safe state is entered by switching to input function with internal pull-up connected or tri-state, depending on the configured reset value of the corresponding Pn\_IOCR register through PMSWCR5.TRISTREQ or setting of HWCFG[6].(the content of the corresponding PCx bit fields in register Pn\_IOCR will not be considered).

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

### Exceptions for Emergency Stop Implementation

The Emergency Stop function is available for all GPIO Ports with the following exceptions:

- Not available for P20.2 (General Purpose Input/GPI only, overlayed with Testmode)
- Not available for P40.x and P41.x (analog input ANx overlayed with GPI)
- Not available for P32.0 and P32.1 when using EVRC regulator.
- Not available for P21.2 (used as EMGSTOPB pin).
- Not available for P33.8 (used as EMGSTOPA pin).
- Not available for dedicated I/O without General Purpose Output function (e.g ESRx, TMS, TCK)

The Emergency Stop function can be overruled on the following GPIO Ports:

- P00.x: Emergency Stop can be overruled by the VADC. Overruling can be disabled via the control register P00\_PCSR.
- P14.0 and P14.1: Emergency Stop can be overruled in the DXCPL (DAP over CAN physical layer) mode. No Overruling in the DXCM (Debug over can message) mode
- P21.6: Emergency Stop can be overruled in JTAG mode if this pin is used as TDI
- P21.7: Emergency Stop can be overruled in JTAG or Three Pin DAP mode.
- P33.0-7, P33.9-15 and P34.1: Emergency Stop can be overruled by the 8-Bit Standby Controller (SCR), if implemented. Overruling can be disabled via the control register P33\_PCSR and P34\_PCSR.

On pins with LVDS TX pads the Emergency Stop affects only the CMOS driver not the LVDS driver. Thus only when LPCRx.TX\_EN selects CMOS mode the output is switched off. When TX\_EN selects LVDS mode the output is not switched off.

### 14.4.14 Port Input Register

#### Port n Input Register

The logic level of a GPIO pin can be read via the read-only port input register Pn\_IN. Reading the Pn\_IN register always returns the current logical value at the GPIO pin independently whether the pin is selected as input or output.

IN															
Port n Input Register (024 <sub>H</sub> ) Application Reset Value: 0000 XXXX <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Px (x=0-15)	x	rh	<b>Input Bit x</b> This bit indicates the level at the input pin Pn.x. 0 <sub>B</sub> The input level of Pn.x is 0. 1 <sub>B</sub> The input level of Pn.x is 1.

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
0	31:16	r	<b>Reserved</b> Read as 0.

#### **14.4.15 Access Protection Registers**

## **Port n Access Enable Register 0**

Each port has its own dedicated ACCEN0 and ACCEN1 registers.

The Access Enable Register 0 controls write<sup>1)</sup> access for transactions with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI\_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 and ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCENO.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B , ... , EN31 -> TAG ID 011111B.

ACCENO

## **Port n Access Enable Register 0**

(OFC<sub>H</sub>)

## **Application Reset Value: FFFF FFFF**

Field	Bits	Type	Description
<b>ENx (x=0-31)</b>	x	rw	<p><b>Access Enable for Master TAG ID x</b></p> <p>This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n</p> <p>0<sub>B</sub> Write access will not be executed</p> <p>1<sub>B</sub> Write access will be executed</p>

**Table 522 Access Mode Restrictions of ACCENO sorted by descending priority**

<b>Mode Name</b>	<b>Access Mode</b>		<b>Description</b>
Supervisor Mode and Safety ENDINIT	rw	ENx (x=0-31)	write access only for masters with supervisor mode
Otherwise (default)	r	ENx (x=0-31)	

## **Port n Access Enable Register 1**

Each port has its own dedicated ACCEN0 and ACCEN1 registers.

The Access Enable Register 1 controls write<sup>1)</sup> access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI\_FPI is

1) The BPI\_FPI Access Enable functionality controls only write transactions to the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers.

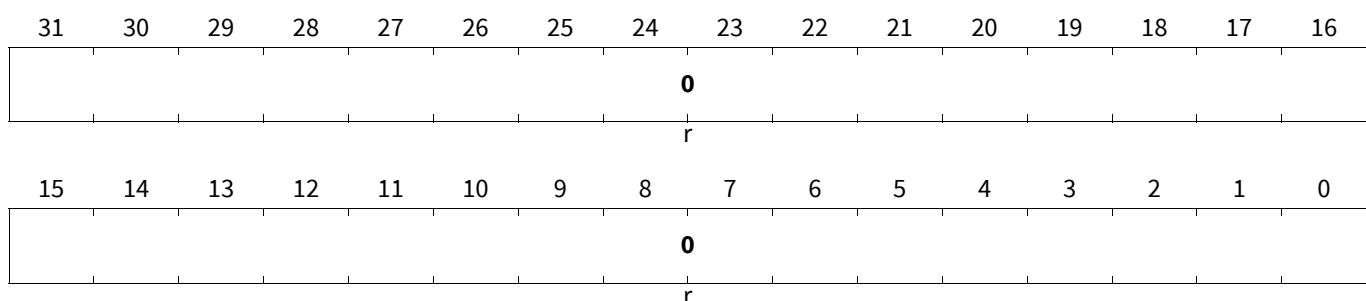
## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in this product.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000B, EN1 -> TAG ID 100001B, ... ,EN31 -> TAG ID 111111B.

### ACCEN1

**Port n Access Enable Register 1** **(0F8<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
0	31:0	r	<b>Reserved</b> Read as 0; should be written with 0

**Table 523 Access Mode Restrictions of ACCEN1 sorted by descending priority**

Mode Name	Access Mode	Description
Supervisor Mode and Safety ENDINIT	-	See bit field definitions above write access only for masters with supervisor mode
Otherwise (default)	-	See bit field definitions above

## General Purpose I/O Ports and Peripheral I/O Lines (Ports)

### 14.5 Revision History

**Table 524 Revision History from V1.8.19 to the latest revision**

Reference	Changes to Previous Version	Comment
<b>V1.8.20</b>		
<a href="#">Page 34</a>	Revision History entries up to V1.8.19 removed.	
<a href="#">Page 8</a>	Removed confusing phrase “, only input selection apply.” from register IOCRx from bitfield description of PC.	
-	Only cosmetic change: register documentation generator merges more reserved bit fields (e.g. “0” or “1” bit fields).	
<b>V1.8.21</b>		
<a href="#">Page 1</a>	Removed “hysteresis” in Feature List from bullet point “Controls pad characteristics of the assigned pads like drive strength, slew rate, pull-up/down, hysteresis, ...”	

---

**Safety Management Unit (SMU)****15      Safety Management Unit (SMU)**

## Safety Management Unit (SMU)

### 15.1 Feature List

The SMU implements the following features:

- Collects every alarm signal generated from safety mechanisms
- Alarm flags are stored in a diagnosis register that is only reset by the Power-on reset, to enable fault diagnosis and possible recovery.
- An alarm emulation facility is provided to enable software-based diagnostics to post an alarm condition with the same properties as the hardware alarms.
- Implements the access protection and Safety ENDINIT modes to protect configuration registers.
- Implements a Fault Signaling Protocol (FSP) reporting internal faults to the external environment. The FSP can be configured using the following modes:
  - Bi-stable single pin output, also called ErrorPin (push-pull active low configuration using SMU\_FSP0)
  - Timed dual rail coding using two inverted values on the ErrorPins (SMU\_FSP0 and SMU\_FSP1)
  - Single-bit timed protocol using the ErrorPin
- The FSP value driven by the microcontroller can be observed via the FSP Status Register.
  - Additionally a monitor is available to check the timing and state properties of the FSP protocol when a fault is reported.
- After power-on reset the FSP is disabled. Software needs to connect the FSP to port using the Port Control Register
- Each individual alarm can be configured to activate the fault signaling protocol.
- Two SMU instance: one located in the core domain called SMU\_core and another in the stand-by domain called SMU\_stby
- Alarms processed in SMU\_core can be configured to activate one of the following internal actions:
  - generate an interrupt request to any of the CPUs, concurrent interrupts to several CPUs can be configured
  - generate a NMI request to the System Control Unit
  - generate a reset request to the System Control Unit
  - activate the Port Emergency Stop signal controlling the safe state of output pads
  - generate a CPU reset request
- All power and temperature related alarms are processed in a diverse way by both the SMU\_core and SMU\_stby.
- Implements an SMU Alive alarm which signals if the SMU\_core is not triggering the configured reaction when an alarm is raised.
- After reset every alarm reaction, except for watchdog time-out alarm, is disabled.
- A lock mechanism is available to protect the SMU configuration
- Implements internal watchdog(s) time-out pre-warning function.
- Implements an internal watchdog called recovery timer to monitor the execution of critical software error handlers. The watchdog is started automatically by hardware according to configurable alarm events.

### 15.2 Overview

The SMU is a central component of the safety architecture providing a generic interface to manage the behavior of the microcontroller under the presence of faults. The SMU centralizes all the alarm signals related to the different hardware and software-based safety mechanisms. Each alarm can be individually configured to trigger internal actions and/or notify externally the presence of faults via a fault signaling protocol. The severity of each alarm shall be configured according to the needs of the safety application(s): per default every alarm reaction is

## Safety Management Unit (SMU)

disabled with the exception of the watchdog time-out alarms. For debug and diagnosis purposes the alarm signals set a sticky bit, which is resilient to application or system resets. The SMU also implements some housekeeping functions related to the management and test of dedicated safety mechanisms. A special test mode is available to test the SMU itself enabling to detect latent faults. In addition to the register access protection, the SMU implements a configuration locking mechanism. Moreover, in order to mitigate the potential common cause faults, the SMU is partitioned in two parts:

- SMU\_core: located in the core domain
- SMU\_stby: located in the stand-by domain

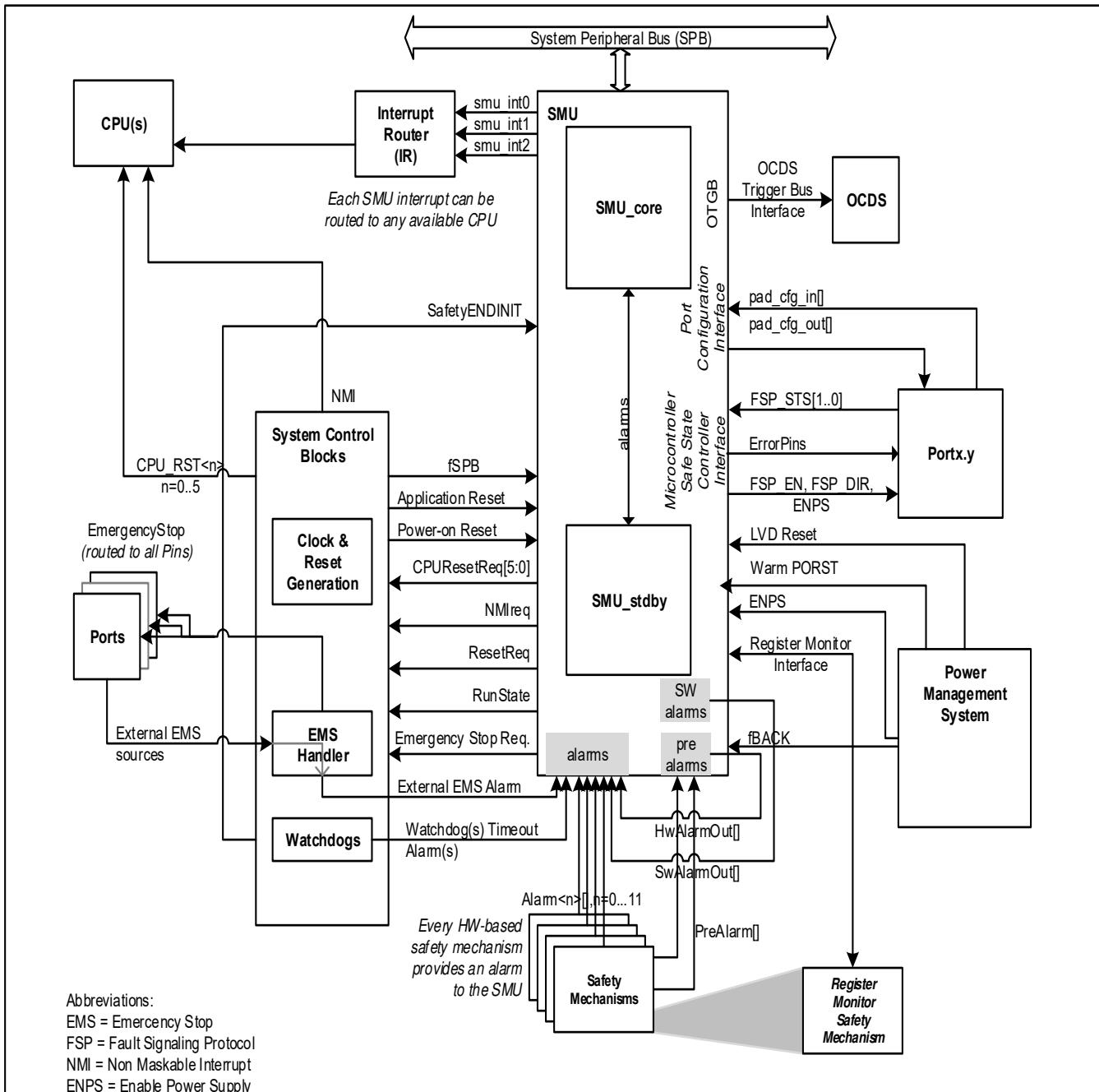
The SMU\_core and SMU\_stby are diverse in the way they are designed and in their timing. There is a physical isolation between the two parts of the SMU. They are located in different clock and power domains. This allows the SMU to process any incoming alarm regardless of the frequency of the clock used to generate this alarm. Also, alarm events generated on fSPB (or derivatives) will be processed by the SMU\_core and alarm events generated on fBACK by the SMU\_stby. This way, all Clock Alive Monitor alarms are processed in the same clock domain as they are generated. Moreover, power and temperature related alarms are processed in a diverse way since they are processed by both SMU\_core and SMU\_stby. One or more reactions to these alarms could be configured in the SMU\_core or the SMU\_stby.

Also, in order to detect errors in the SMU\_core an alarm, **smu\_core\_alive**, is sent from the SMU\_core to the SMU\_stby. The reaction to these alarms is configurable in both domains. However, for the SMU\_stby, only no reaction or setting the ErrorPins in high impedance state can be configured as an alarm reaction.

The SMU in combination with the embedded safety mechanisms enable to detect and report more than 99% of the critical failure modes of the microcontroller within the fault tolerance time interval. The timing characteristics of the fault tolerance time interval can be configured in the SMU.

**Figure 157** gives an overview of the SMU interfaces.

## Safety Management Unit (SMU)



**Figure 157 SMU Interfaces**

### 15.2.1 Architecture

Figure 158 gives an overview of the SMU architecture.

## Safety Management Unit (SMU)

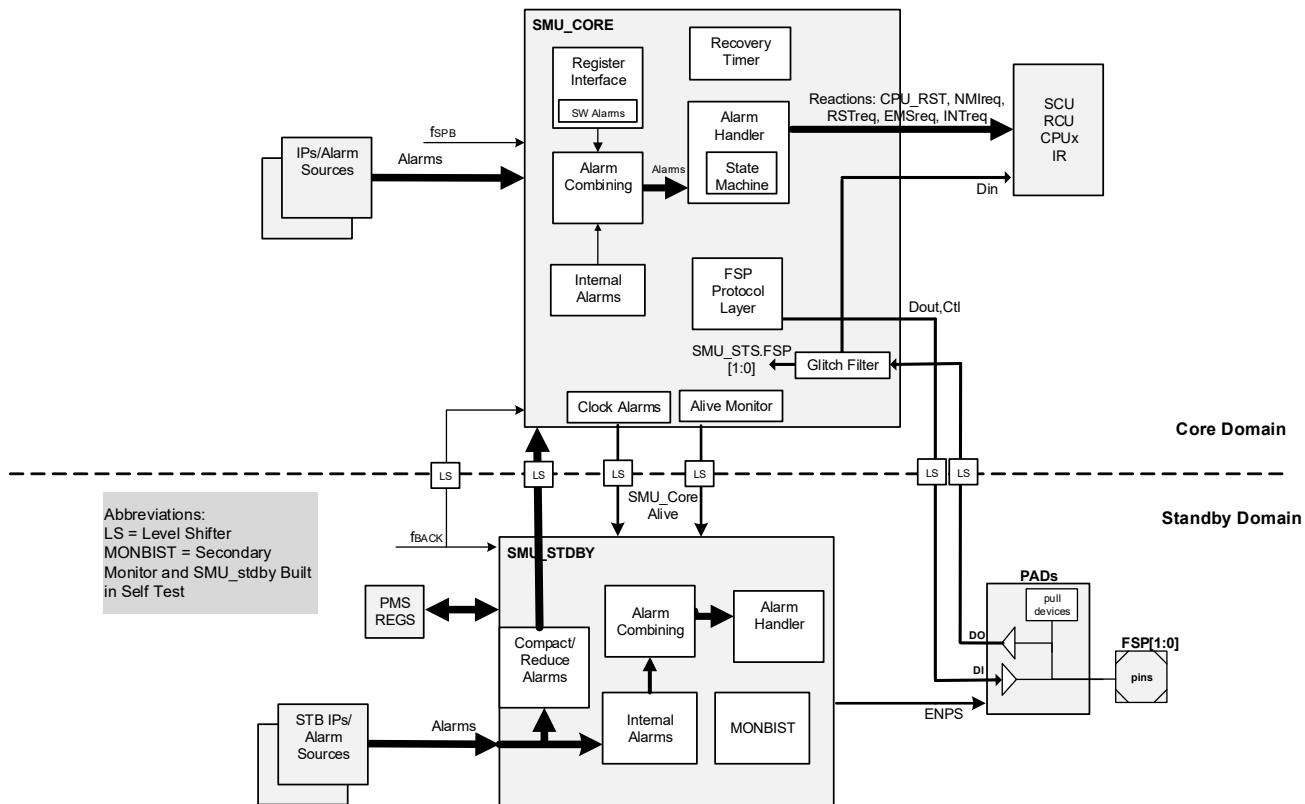


Figure 158 SMU Architecture

### 15.2.2 SMU\_core

The core domain SMU, also called SMU\_core, collects the majority of the alarms signals from the hardware monitors, safety mechanisms, defined by the safety concept. The section **Alarm Mapping** specifies the alarm interface and classifies them into alarm groups. The alarm groups define a logical mapping used to map alarm signals to internal status registers. The section **Alarm Handling** describes the configuration options. The configuration options specify the behavior of the SMU\_core when an alarm event is detected. The alarm event can trigger an internal action and/or the activation of the ErrorPin(s) that indicates the presence of a fault to the external environment. The section **SMU\_core Control Interface** specifies how the SMU\_core can be controlled by software and the dependencies with the hardware operation. The section **Fault Signaling Protocol (FSP)** describes the properties of the external fault signaling protocol defining the timing and logical properties of the ErrorPin(s).

### 15.2.3 SMU\_stdby

The stand-by domain SMU, also called SMU\_stdby, collects alarms from modules which detect clock (no clock), power (under/over voltage) and temperature failures (under/over temperature). The SMU\_stdby also collects the SMU\_alive alarm signal which notifies when the SMU\_core is not triggering a reaction after an alarm is raised. Moreover, the SMU\_stdby implements a **Built-In Self Test** feature which allows users to test the SMU\_stdby reaction to alarm signals and the complete alarm path from the Secondary Voltage Monitor to the SMU\_stdby. Please refer to the Power Management System chapter for more details on the Secondary Monitor and SMU\_stdby Built in Self Test.

The section **Alarm Mapping** specifies the alarm interface and classifies them into two alarm groups. The section **Alarm Handling** describes the configuration options that can be specified. The configuration options specify the behavior of the SMU\_stdby when an alarm event is detected. The alarm event can trigger the activation of the ErrorPins that indicates the presence of a fault to the external environment.

## Safety Management Unit (SMU)

### 15.3 Functional Description

This section describes the SMU\_core, the SMU\_stby and the interdependencies between them.

#### 15.3.1 SMU\_core

##### 15.3.1.1 Reset Types

The SMU\_core requires multiple reset types. The reset types are fully specified in the System Control Units. The reset types that are required by the SMU\_core are:

- Power-on Reset
- System Reset
- Debug Reset
- Application Reset

**Table 525** specifies the scope of each reset type to the SMU\_core control configuration and logic.

**Table 525 Effect of Reset Types to SMU functionality**

SMU Function	Application Reset	Debug Reset	System Reset	Power-on Reset
SMU_core FSP Function <a href="#">Chapter 15.3.1.8</a>	Not Affected	Not Affected	Not Affected	Reset
SMU_core State Machine Function <a href="#">Chapter 15.3.1.7</a>	Not Affected	Not Affected	Not Affected	Reset
SMU_core Debug Function <a href="#">Chapter 15.3.1.9</a>	Not Affected	Reset	Not Affected	Reset
SMU_core Alarm Diagnosis Registers <a href="#">Chapter 15.4.1.6</a>	Not Affected	Not Affected	Not Affected	Reset
SMU_PCTL.PCS Register Field <a href="#">PCTL</a>	Not Affected	Not Affected	Not Affected	Reset
SMU_core Alive Monitor <a href="#">Chapter 15.3.1.2.5</a>	Reset	Not Affected	Reset	Reset
SMU_core Glitch Filter <a href="#">Chapter 15.3.1.2.3</a>	Reset	Not Affected	Reset	Reset
SMU_core SPB BPI	Reset	Not Affected	Reset	Reset
SMU_core Other Functions	Reset	Not Affected	Reset	Reset

#### 15.3.1.2 Interfaces Overview

This section describes the interface signals between the SMU\_core and other modules.

##### 15.3.1.2.1 Interfaces to SCU

Internal actions resulting from an alarm event that interface to the System Control Unit. The interface signals are:

- Emergency Stop Request
- Reset Request
- NMI Request

## Safety Management Unit (SMU)

- CPU Reset Request

### 15.3.1.2.2 Interfaces to the Interrupt Router

Internal actions resulting from an alarm event that interface to the Interrupt Router. The interface signals are:

- SMU Interrupt Service Request 0
- SMU Interrupt Service Request 1
- SMU Interrupt Service Request 2

The mapping of SMU Interrupt Service Requests to the Interrupt Router (IR) interrupt nodes can be found in the Interrupt Router chapter (SRC\_SMUy, y=0..2).

The **AGC.IGCSx**, x={0,1,2} register fields provide the software interface to control how the SMU triggers interrupt requests to the interrupt router.

Each **AGC.IGCSx** is a 3-bits bit-field:

- **AGC.IGCSx[0]** shall be set to ‘1’ to trigger SMU Interrupt Service Request 0
- **AGC.IGCSx[1]** shall be set to ‘1’ to trigger SMU Interrupt Service Request 1
- **AGC.IGCSx[2]** shall be set to ‘1’ to trigger SMU Interrupt Service Request 2

The usage of the three **AGC.IGCSx** bit-fields is defined in [Alarm Configuration](#);

### 15.3.1.2.3 Interface to the Ports (ErrorPin)

The generic port structure is presented in presented in [Figure 159](#).

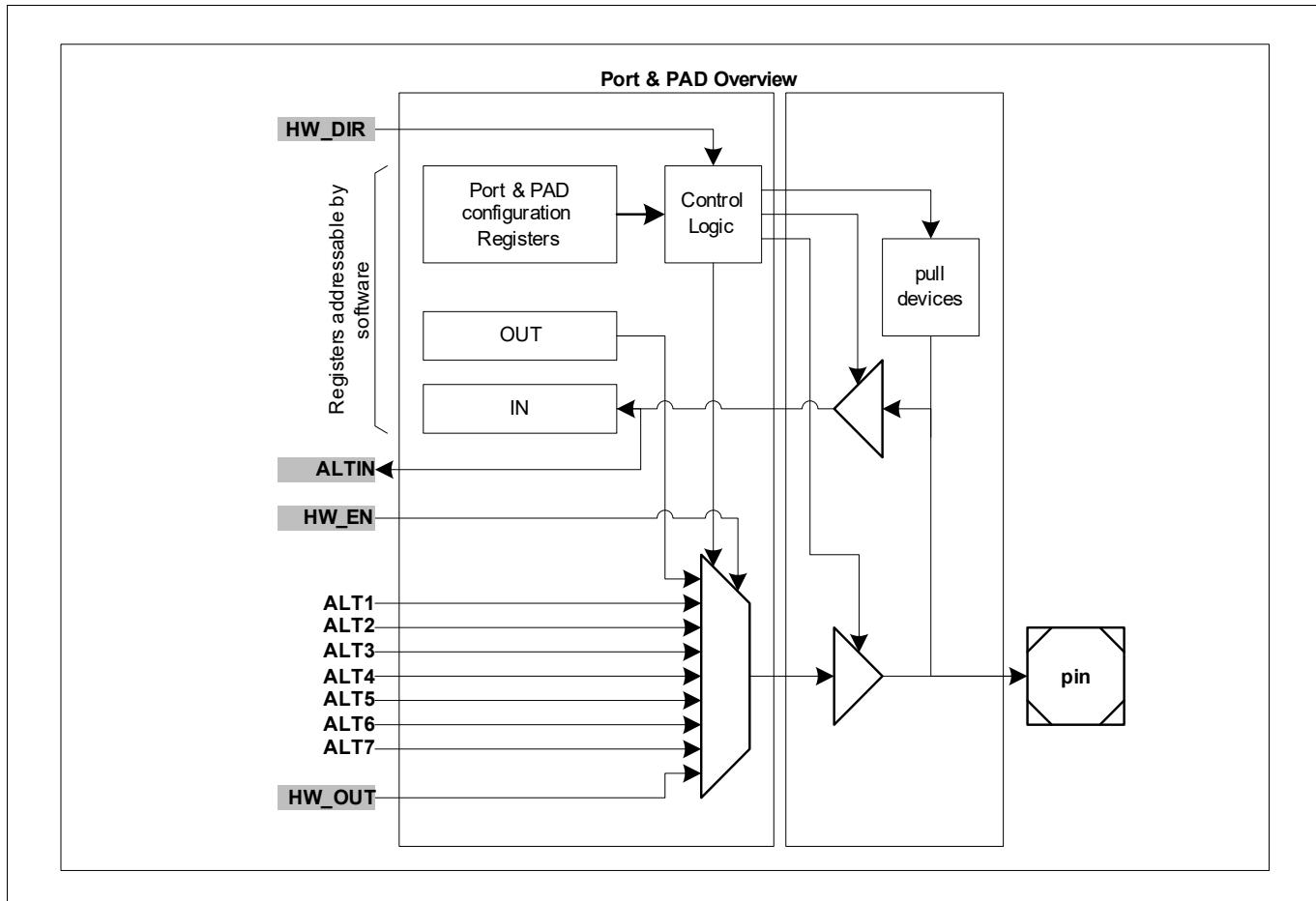
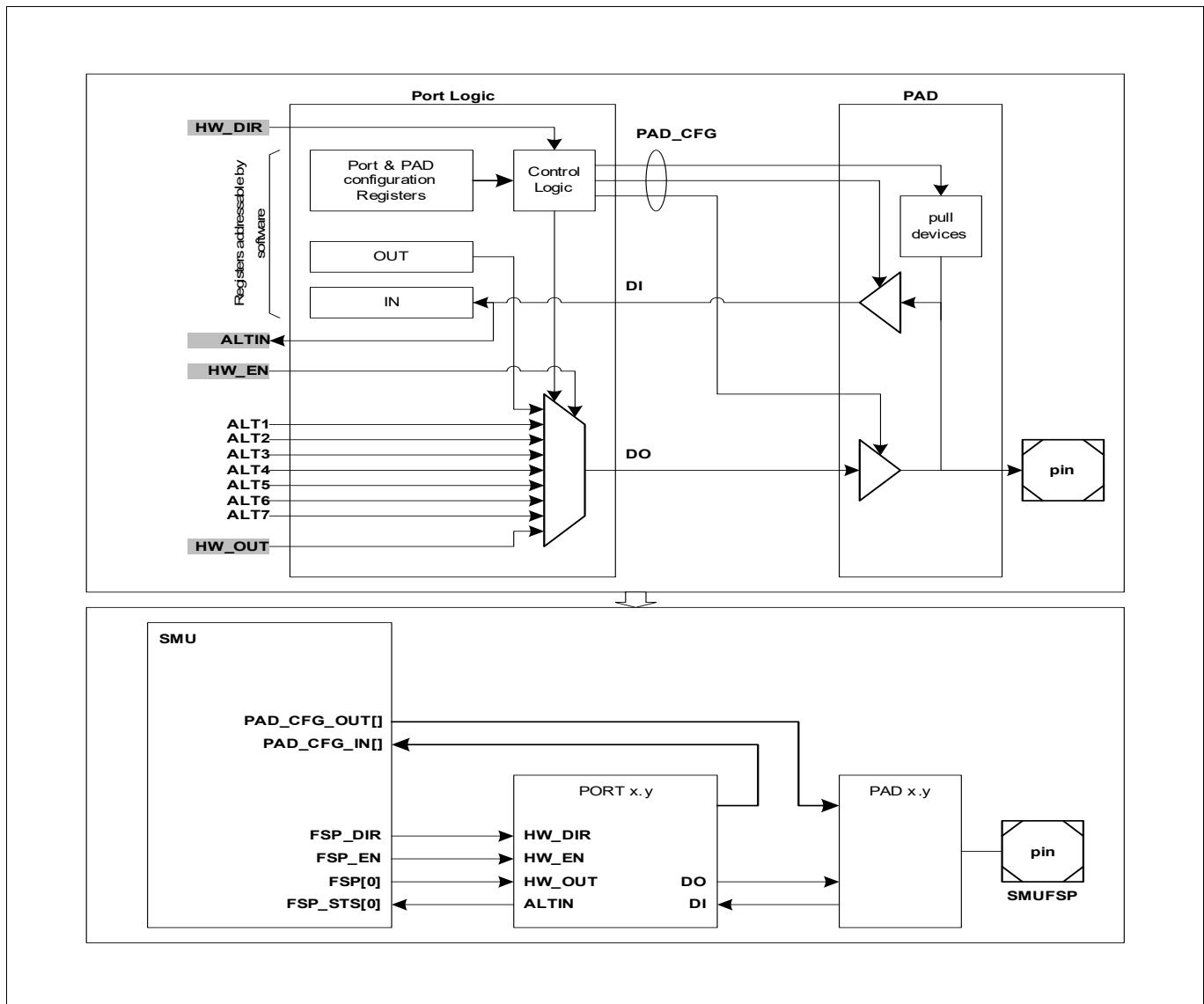


Figure 159 Generic Port Structure

## Safety Management Unit (SMU)

The port pin can be connected to peripheral via the ALTx output lines. This is the default state of the port after power-on reset (see Ports chapter for more details). The SMU\_core connects to the port using the HW\_DIR, ALTIN, HW\_EN, HW\_OUT signals. When the HW\_EN port input is driven active by the SMU\_core, SMU\_core gets full control over the port, bypassing any other software configuration related to the usage of the ALTx inputs.

**Figure 160** provides a more detailed overview of the port structure and highlights the signals involved in the SMU\_core connectivity.



**Figure 160** SMU/PAD Control Interface to the PADs

SMU\_FSP0 (FSP[0] in **Figure 160**) is controlled by hardware. FSP\_DIR and FSP\_EN are controlled by software as follows:

- FSP\_DIR output is directly driven by the **PCTL.HWDIR**
- FSP\_EN output is directly driven by the **PCTL.HWEN**

This also applies for SMU\_FSP1 when used.

**PCTL** provides a field PCS that, in combination with the P33\_PCSR.SEL bitfield, enables software to change the PAD control of FSP[0]. With P33\_PCSR.SEL, **PCTL** HWDIR, HWEN and PCS fields, software can control the FSP[0] PAD ownership transition from GPIO to full SMU hardware control.

## Safety Management Unit (SMU)

SMU\_FSP1 PAD is not under the control of the SMU\_core. **PCTL** PCS field does not enable software to change the PAD control of SMU\_FSP1. However, the **PCTL** HWDIR and HWEN fields can be used to overrule the PAD configuration.

The contents of the **PCTL** register are locked by the **KEYS** register and are only reset by a power-on reset, therefore the PAD configuration remains preserved even in the presence of an application or system reset. Furthermore the **PCTL** register is implemented using safety flip-flops safety mechanism that detects, during run-time, any bit change caused by a random hardware fault.

Refer to [SMU\\_core Integration Guidelines](#) for the SMU and PORT configuration steps that are required to use the ErrorPin.

### Glitch Filter (not available in TC39x A-Step)

In systems which are using the ErrorPin in Open Drain mode, glitches up to 1.2 µs can be suppressed by a glitch filter. There are two relevant paths from the ErrorPin in case of Open Drain mode usage:

- ErrorPin to **STS.FSP[0]**
  - For this path the filter can be switched on/off in **PCTL.GFSTS\_EN**
- ErrorPin to SCU for Port Emergency Stop usage
  - For this path the filter can be switched on/off in **PCTL.GFSCU\_EN**

## Safety Management Unit (SMU)

### 15.3.1.2.4 Interface to the Register Monitor

#### Safety Flip-flop Self-Test Protocol

The interface between the Register Monitor Control (**RMCTL**), Register Monitor Error Flag (**RMEF**) and Register Monitor Self Test Status registers (**RMSTS**) is specified as follows:

- **RMCTL.TE[31:0]**
  - Setting **RMCTL.TE[i]** to 1 starts a self-test on the safety flip-flop protected registers of a given module (see **Table 526**)
  - These bits have to be set back to 0 at the end of the self-test
- **RMEF.EF[31:0]**
  - **RMEF.EF[i]** is set to 1 whenever a fault is detected in a safety flip-flop protected register of a given module, regardless of the state of the **RMCTL.TE[i]** (see **Table 527**)
- **RMSTS.STS[31:0]**
  - **RMSTS.STS[i]** is set to 1 at the end of a safety flip-flop self-test sequence (see **Table 528**)

The mapping of the Register Monitor registers is specified as follows:

**Table 526 Register Monitor Self-Test Enable**

SMU_RMCTL	Module
SMU_RMCTL[0]	MTU
SMU_RMCTL[1]	IOM
SMU_RMCTL[2]	IR
SMU_RMCTL[3]	EMEM
SMU_RMCTL[4]	SCU/SRU
SMU_RMCTL[5]	PMS
SMU_RMCTL[6]	DMA
SMU_RMCTL[7]	SMU_core
SMU_RMCTL[8]	CERBERUS
SMU_RMCTL[9]	SYS_PLL/PER_PLL
SMU_RMCTL[10]	CCU
SMU_RMCTL[31..11]	Reserved

**Table 527 Register Monitor Self-Test Fail**

SMU_RMEF	Module
SMU_RMEF[0]	MTU
SMU_RMEF[1]	IOM
SMU_RMEF[2]	IR
SMU_RMEF[3]	EMEM
SMU_RMEF[4]	SCU/SRU
SMU_RMEF[5]	PMS
SMU_RMEF[6]	DMA

## Safety Management Unit (SMU)

**Table 527 Register Monitor Self-Test Fail (cont'd)**

<b>SMU_RMEF</b>	<b>Module</b>
SMU_RMEF[7]	SMU_core
SMU_RMEF[8]	CERBERUS
SMU_RMEF[9]	SYS_PLL/PER_PLL
SMU_RMEF[10]	CCU
SMU_RMEF[31..11]	Reserved

**Table 528 Register Monitor Self-Test Done**

<b>SMU_RMSTS</b>	<b>Module</b>
SMU_RMSTS[0]	MTU
SMU_RMSTS[1]	IOM
SMU_RMSTS[2]	IR
SMU_RMSTS[3]	EMEM
SMU_RMSTS[4]	SCU/SRU
SMU_RMSTS[5]	PMS
SMU_RMSTS[6]	DMA
SMU_RMSTS[7]	SMU_core
SMU_RMSTS[8]	CERBERUS
SMU_RMSTS[9]	SYS_PLL/PER_PLL
SMU_RMSTS[10]	CCU
SMU_RMSTS[31..11]	Reserved

### Safety Flip-flop Self-Test Conditions

In order prevent unexpected behaviors during the self-test, the following conditions shall to be fulfilled:

- The clock of the module to be tested must be enabled
- The clocks and clock ratios of the modules involved in the self-test shall not be modified during the self-test
- The clock ratio of the modules involved in the self-test sequence shall be set in a specific way. For more detail please refer to the safety flip-flop related chapter
- The microcontroller shall not be set in reset, sleep or debug state whilst a self-test is being executed

### 15.3.1.2.5 Interface to SMU\_stdby

In case of a malfunction the SMU\_core generates a signal, smu\_core\_alive, to the SMU\_stdby.

The smu\_core\_alive signal will be generated if one of the following conditions is meet:

- An alarm event occurs while the SMU\_core is in RUN or FAULT state and the SMU\_core Alive Monitor, SCAM, detects that a reaction has not been generated by the SMU\_core.
- A watchdog or recovery timer alarm event occurs while the SMU\_core is in START state and the SCAM detects that a reaction has not been generated by the SMU\_core.
- SMU\_ActivateFSP or SMU\_ActivatePES command is sent but the appropriate reaction is not generated by the SMU\_core
- An alarm's configuration is changed while this alarm is being processed

---

## Safety Management Unit (SMU)

The smu\_core\_alive signal can be tested by sending the SMU\_AliveTest command.

Indeed, sending the SMU\_AliveTest command will trigger the SCAM to inject a fault and to forward the smu\_core\_alive alarm to the SMU\_stdby. The smu\_core\_alive alarm flag can be read in the [AG2i\\_STDBY \(i=1\)](#) whether or not the SMU\_stdby is enabled (see [Interdependency Between SMU\\_core and SMU\\_stdby](#)). The SCAM error injection can be disabled by sending the SMU\_AliveTest command with a different argument (see [Table 534](#)). However, the SMU\_AliveTest command cannot clear the smu\_core\_alive alarm when this one is generated by a real fault. An application reset (at least) is needed to clear the smu\_core\_alive alarm.

For the TC39xB and the TC38x, when the SMU\_core is in START state, a fault in the processing of the Recovery Timer 1 timeout alarm will not generate the smu\_core\_alive alarm.

## Safety Management Unit (SMU)

### 15.3.1.3 SMU\_core Integration Guidelines

This chapter extends the [Interfaces Overview](#) section by providing additional information for the usage of the ErrorPin (**Fault Signaling Protocol (FSP)**) in combination with other input/output (GPIO) functions of the microcontroller and the configuration of the **Fault Signaling Protocol (FSP)**.

**Note:** *The PAD properties (push-pull, open-drain, drive strength,...) are configured in the registers of the PORT to which the SMU connects to. These registers are described in the Ports chapter.*

- During power-on-reset, the ErrorPin is in high impedance: the pull devices are disabled.
- After power-on-reset the default mode of the PORT to which the ErrorPin is connected is GPIO.
- Before changing the ownership of the PAD to SMU, software shall configure the PORT registers including the following:
  - Disable the pull devices if GPIO is not used
  - Program the GPIO registers of the ErrorPin to strong driver output constant low
  - Set P33\_PCSR.SEL8 to 1
  - Set P33\_PCSR.SEL10 to 0
- To enable SMU to control the ErrorPin PAD, software shall activate the PAD configuration safeguarding process (see [Interface to the Ports \(ErrorPin\)](#)).
  - The safeguarding process requires a software action that consists in writing a 1 into the **PCTL.PCS** field.  
**Only the first transition from 0 to 1 leads to the safeguarding process. A new PORT configuration followed by a new transition from 0 to 1 of the PCTL.PCS has no effect on the hardware.**

Also, the following steps need to be followed to reconfiguring the **Fault Signaling Protocol (FSP)** settings:

- While in Fault Free State and the Time Switching or Dual Rail protocol is in use:
  - Disable Time Switching or Dual Rail protocol by setting FSP in Bi-stable protocol mode (FSP.MODE = 00B)
  - Wait until Bi-stable protocol mode is active (read back register FSP twice)
  - Write desired value to PRE1, PRE2 or TFSP\_HIGH
  - Switch FSP.MODE to the desired protocol (if required)
- While in Fault Free State and the Bi-stable protocol is in use:
  - Write desired value to PRE1, PRE2 or TFSP\_HIGH
  - Switch FSP.MODE to the desired protocol (if required)

---

## Safety Management Unit (SMU)

### 15.3.1.4 Alarm Mapping

Please refer to the appendix document for device specific alarm tables

#### 15.3.1.4.1 SMU\_core Internal Alarms

The following tables describe the alarms generated by the SMU\_core

##### SMU\_CORE Alarm Table

**Table 529 SMU\_CORE Alarm Table**

Description	SMU Targets
SMU.SMU_CORE - SMU_core Alive Alarm	ALM21[16]
SMU_core - Safety flip-flop uncorrectable error	ALM6[7]
SMU_core - Safety flip-flop uncorrectable error	ALM10[21]
SMU.SMU_core - Recovery Timer 0 Time-out alarm	ALM10[16]
SMU.SMU_core - Recovery Timer 1 Time-out alarm	ALM10[17]

##### FSP Alarm Table

**Table 530 FSP Alarm Table**

Description	SMU Targets
SMU.SMU_core - ErroPin Fault State Activation alarm	ALM10[18]

## Safety Management Unit (SMU)

### 15.3.1.5 Alarm Handling

This section specifies the hardware and software alarm processes.

#### 15.3.1.5.1 Alarm protocol

Each safety mechanism shall interface with the SMU\_core using a pre-defined protocol. The protocol enables to cross clock domains in a reliable manner. The operation of the protocol has no influence to the software layers.

#### 15.3.1.5.2 Alarm Configuration

Upon reception of an alarm event the SMU\_core decodes the actions to be performed. The action can be classified into an internal behavior and an external behavior. Both the internal and external behavior can be configured for every alarm.

The external behavior is related to the Fault Signaling Protocol (see [Fault Signaling Protocol \(FSP\)](#)). The external behavior is configured via the following registers:

- [AGiFSP \(i=0-11\)](#)

The internal behavior of the SMU under the presence of an alarm is controlled via the following registers:

- [AGiCFj \(i=0-11;j=0-2\)](#)

The internal behavior is specified by a 3-bit code as follows:

- Code = SMU\_AG<n>CF2. SMU\_AG<n>CF1. SMU\_AG<n>CF0, n=0...11

**Table 531 SMU Alarm Configuration**

Code	Name	Behavior
0x0	SMU_NA	No Action. Reset value. Alarm disabled.
0x1	SMU_RSVD	Reserved. No Action. Alarm disabled.
0x2	SMU_IGCS0	Sends an interrupt request to the interrupt system according to the Interrupt Generation Configuration Set 0 from the <a href="#">AGC</a> register.
0x3	SMU_IGCS1	Sends an interrupt request to the interrupt system according to the Interrupt Generation Configuration Set 1 from the <a href="#">AGC</a> register.
0x4	SMU_IGCS2	Sends an interrupt request to the interrupt system according to the Interrupt Generation Configuration Set 2 from the <a href="#">AGC</a> register.
0x5	SMU_NMI	Sends an NMI request to the SCU
0x6	SMU_RESET	Sends a reset request to the SCU. The SCU shall be configured to generate an application or system reset.
0x7	SMU_CPU_RST	Triggers a CPU reset request using CPU Reset Configuration Set from the <a href="#">AGC</a> register

#### 15.3.1.5.3 Alarm operation

Whenever an input alarm event is detected and the SMU\_core state machine is in the RUN or FAULT state, the SMU checks for the corresponding actions to be done for the internal action and for the FSP in a concurrent way. If an input alarm event is detected and no action is specified for the alarm, the corresponding status bit shall be set to 1 as well but no action takes place.

The processing of the incoming alarm events is performed as follows:

- All alarm groups and every alarm are scanned at the same time.

## Safety Management Unit (SMU)

- The execution of pending alarms is done concurrently.
- The processing of an alarm within an alarm group may take several fSPB cycles.
- If a fault handling is done, the corresponding bit in SMU\_AEX register is set. As long as a bit is set, the corresponding fault handling is blocked. This bit needs to be reset by SW after fault handling is done.
  - If the alarm execution bit in SMU\_AEX register related to a pending alarm is already set, then the alarm event is ignored but the status bit and also the corresponding alarm missed event bit are set.
  - If the status flag related to an alarm event is already set to 1, the alarm event is ignored.
  - Whenever an alarm event is processed, the corresponding status bit is set to 1 by hardware in the AG<x> register. If an internal SMU\_core action is configured and executed, the action counter (ACNT) in the **AFCNT** register increments.

## Safety Management Unit (SMU)

### 15.3.1.5.4 Alarm Status Registers

**Table 532** specifies the possible software actions on the AG<x> alarm group status registers depending on the SMU\_core State machine state.

**Table 532 Handling of Alarm Status**

SMU State Machine	SW Action	Effect on AG<x>
START	SMU_ASCE(0) command Write Data at AG<x> Address	If (Data[i] == 1 && (ALARM[i] == 0    (ALARM[i] != WDTx_ALARM && ALARM[i] != RTx_ALARM <sup>1)</sup> ))) AG<x>[i] = 0 else “no effect”
START	Write Data at AG<x> Address	If (Data[i] == 1) AG<x>[i] = 1 else “no effect”
RUN or FAULT	SMU_ASCE(0) command Write Data at AG<x> Address	If (Data[i] == 1 && ALARM[i] == 0) AG<x>[i] = 0 else “no effect”
RUN or FAULT	Write Data at AG<x> Address	No Effect

- 1) Recovery timer alarms are not clearable in START state only if when they have their default alarm configuration. See [Chapter 15.3.1.5.7](#) for more detail about recovery timers default configuration.

In the START state software has the possibility to “emulate” the occurrence of input alarm events by writing at an AG<x> address. Software shall read back the AG<x> register to ensure the completion of the operation if necessary. Also, after clearing an alarm, software shall re-check the alarm status bit (for alarms occurring during the time window of the clearing).

To clear individual alarm flags, use only 32bit writes.

### 15.3.1.5.5 Alarm Diagnosis Registers

The alarm diagnosis registers enable the application to improve the diagnosis of the root cause that lead to a malfunction. In that context they may help to implement recovery strategies, if allowed by the application. The SMU\_Ad<sub>x</sub> diagnosis registers shall make a snapshot of the SMU\_AG<sub>x</sub> registers when:

- the action to be executed by the SMU is a reset when the SMU is in the RUN or in the FAULT state
- a condition which switches SMU\_core state machine (SSMSSM) to the FAULT state (RUN -> FAULT, FAULT -> FAULT) takes place, either controlled by the SMU hardware or a software command

The SMU\_Ad<sub>x</sub> registers shall only be cleared by a power-on reset.

**Note:** After every condition which triggers the SMU to make a snapshot of the SMU\_AG<sub>x</sub> registers the SMU\_Ad<sub>x</sub> diagnosis registers are overwritten with the current SMU\_AG<sub>x</sub> register values. This is also valid if the SMU is already in FAULT state and the FSP is activated again.

### 15.3.1.5.6 Port Emergency Stop

The port emergency stop feature enables forcing a pad into General Purpose Input Mode. The port emergency stop request to the SCU can be activated by any of the following situations:

- a SMU\_ActivatePES() software command
- an alarm event with SMU\_AG<x>FSP enabled and **FSP.PES** enabled
- an alarm event with an internal action configured in SMU\_AG<x>CFx registers and SMU\_AGC.PES enabled for that action.

## Safety Management Unit (SMU)

### 15.3.1.5.7 Recovery Timer

A recovery timer (RT) is available to enable the monitoring of the duration or internal error handlers activated by an alarm, NMI or Interrupt action. In the current SMU\_core implementation two independent instances (RT0 and RT1) are available. The recovery timer duration (identical for all instances) is configured in the register **RTC**. It is possible to enable or disable each instance, however both instances are enabled by default as it is required for the operation of the CPU watchdogs (see also [Watchdog Alarms](#)). In addition to **RTC** additional configuration registers (**RTAC00**, **RTAC01**, **RTAC10** and **RTAC11**) are available per recovery timer instance to configure the alarm mapping.

The alarm mapping consists of a pair of parameters {GIDI, ALIDI} (with i = 0..3), where GIDI is a group identifier and ALIDI is the alarm identifier belonging to the group. Four {GIDI, ALIDI} pairs can be configured per recovery timer instance. It is possible to configure the same group identifier several times. If less than four alarms need to be mapped to a recovery timer, the same {GIDI, ALIDI} shall be configured several times.

*Note:* *The use of the recovery timer only makes sense if the internal action is an interrupt or NMI. However no hardware check is done, it is up to software to configure the SMU\_core in the appropriate way.*

If a recovery timer is enabled and for any of the {GIDI, ALIDI} pairs an alarm event occurs and if an internal action is configured leading to an internal action (the alarm status shall be cleared), the recovery timer is automatically started by hardware. Such situation is called a recovery timer event. An alarm without internal action shall not start a recovery timer.

Once a recovery timer event has occurred, the recovery timer starts and counts until software stops it with the `SMU_RTStop()`. If the timer expires, an internal SMU alarm (Recovery Timer Timeout) is issued. During the time the recovery timer is running, any other action that requests the recovery timer is ignored. If such event happens, the bit RTME (Recovery Timer Missed Event) is set to '1' by hardware in the **STS** register. The bit RTME can only be cleared by software. The bit RTS (Recovery Timer Status) is set to '1' by hardware in the **STS** register during the time the recovery timer is running: from the timer activation until a `SMU_RTStop()` is received or the timer expires. The bit RTS is cleared by hardware upon reception of `SMU_RTStop()` or the timer expires.

If a `SMU_RTStop()` command is received when the recovery timer is not active, the command returns an error response.

*Note:* *If RTC.RTD shall be written, make sure no recovery timer is running (recovery timer state is indicated by bits RTS0 and RTS1 in the STS register).*

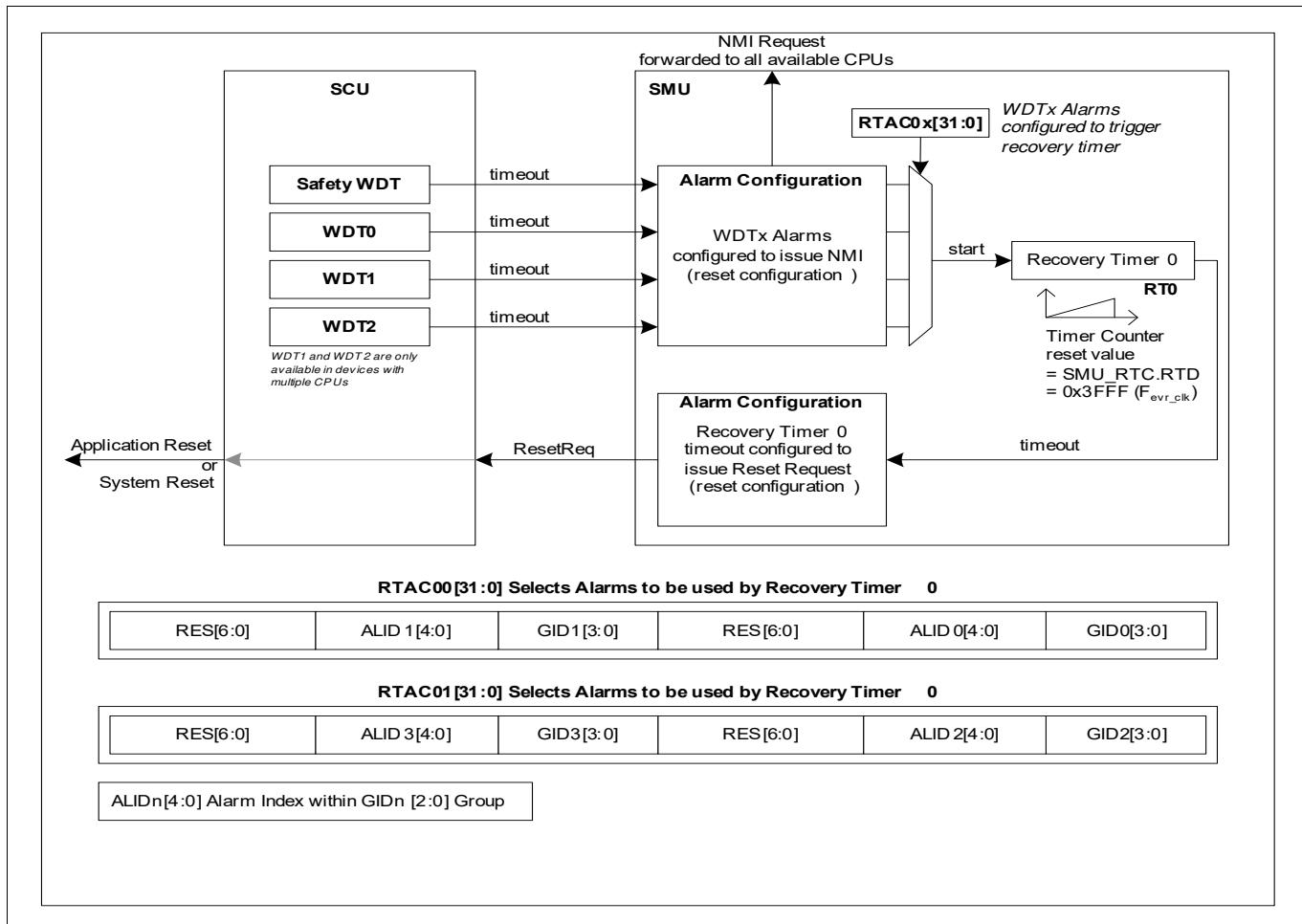
### 15.3.1.5.8 Watchdog Alarms

The watchdogs (WDT) timeout alarms require a special processing in order to ensure a correct microcontroller behavior if the watchdogs are not serviced by software or firmware. It shall be ensured that the microcontroller is reset after a pre-warning phase, where software can still perform some critical actions.

- Every timeout alarm shall activate an NMI
- Recovery Timer 0 shall be configured to service WDT timeout alarms for Safety WDT, CPU0 WDT, CPU1 WDT and CPU2 WDT
- Recovery Timer 1 shall be configured to service WDT timeout alarms for CPU3 WDT, CPU4 WDT and CPU5 WDT
- Recovery Timer 0 and Recovery Timer 1 timeout alarms shall be configured to issue a reset request and activate the Fault Signaling Protocol.

The aforementioned properties are implemented as reset values for the watchdog(s) timeout alarm(s) and for the recovery timer 0 and 1.

## Safety Management Unit (SMU)



**Figure 161 Watchdog timeout alarm configuration (RTAC 0)**

The figure shows only the example for RTAC 0 and the related four WDT. In the same way the RTAC 1 with the related WDT3, WDT4 and WDT5 is set up.

Because the watchdog timeout detection is also required from the very first instruction executed by a CPU, the SMU shall process any watchdog timeout alarm during the START state.

**Note:** If the same behavior is expected for all WDT alarms, it is recommended to use the global WDT timeout alarm that implements a logical OR among all WDT timeout alarms, thus freeing some {GIDI<sub>i</sub>, ALIDI<sub>i</sub>} configuration pairs in **RTAC00**, **RTAC01**, **RTAC10**, and **RTAC11**, for other purposes.

### 15.3.1.6 SMU\_core Control Interface

The core functionality of the SMU\_core is introduced through its control interface. The control interface defines how the SMU\_core can be controlled by software, as summarized in **Table 533**. The control interface is directly linked to the SMU\_core state machine (SSM) operation described in **SMU\_core State Machine** and to the Fault Signaling Protocol (FSP) described in **Fault Signaling Protocol (FSP)**. The control interface is implemented by the **CMD** register using the **CMD** and **ARG** fields. The command completion status is available via the **STS** register.

## Safety Management Unit (SMU)

**Table 533 SMU\_core Commands**

Command	Description	Code
SMU_Start(ARG)	Forces the SSM to go to the RUN state from the START state. Argument ARG shall be set to 0.	0x0
SMU_ActivateFSP(ARG)	Activates the Fault Signaling Protocol. This action is possible in any state of the SSM. Argument ARG shall be set to 0.	0x1
SMU_ReleaseFSP(ARG)	Turns the FSP into the inactive fault free state. In the START state, SMU_ActivateFSP() and SMU_ReleaseFSP() can be called as many times as necessary to perform self tests of every alarm source. Argument ARG shall be set to 0.	0x2
SMU_ActivatePES(ARG)	Triggers the activation of the Port Emergency Stop (PES). The PES is also directly controlled by the SMU_core when entering the FAULT state. Argument ARG shall be set to 0.	0x3
SMU_RTStop(ARG)	Stop the recovery Timer. Argument ARG shall be set to the recovery timer instance available in the product.	0x4
SMU_ASCE(ARG)	Alarm Status Clear Enable Command. Software shall execute this command prior to clear a AG<n> alarm status bit. This command sets the ASCE bit in the <b>STS</b> register. Argument ARG shall be set to 0.	0x5
SMU_Alarm(ARG)	Triggers a software based alarm. ARG specifies the alarm index according to the mapping defined in <b>Alarm Mapping</b> . A software alarm has the same properties as an hardware alarm.	0x6
SMU_AliveTest(ARG)	Enables the testing of the smu_core_alive signal. Sending this command will forward the smu_core_alive alarm to the SMU_stby. Argument ARG shall be set to 0x5 to start the test and to 0xA to end the test.	0x7

Note: *If the argument does not comply with the specification of the command the command is ignored and returns an error code.*

The next table provides the legal conditions for the execution of the commands. The conditions depend on the SMU\_core state machine (SSM) states (see **SMU\_core State Machine**). Any situation not specified leads to an error code.

**Table 534 SMU\_core commands and valid conditions**

Command	SSM state	Other conditions
SMU_Start(ARG)	START	ARG == 0
SMU_AliveTest(ARG)	START	ARG == 0x5 to start the test and 0xA to end the test
SMU_ActivateFSP(ARG)	Any	ARG == 0
SMU_ReleaseFSP(ARG)	START	ARG == 0
SMU_ReleaseFSP(ARG)	FAULT	ARG == 0 & <b>AGC.EFRST == 1<sup>1)</sup></b>
SMU_ActivatePES(ARG)	Any	ARG == 0

## Safety Management Unit (SMU)

**Table 534 SMU\_core commands and valid conditions (cont'd)**

Command	SSM state	Other conditions
SMU_RTStop(ARG)	Any	ARG >= 0 and ARG <= Number of Recovery Timer Instances and Recovery Timer Enabled
SMU_ASCE(ARG)	Any	ARG == 0
SMU_Alarm(ARG)	RUN, FAULT <sup>2)</sup>	ARG >= 0

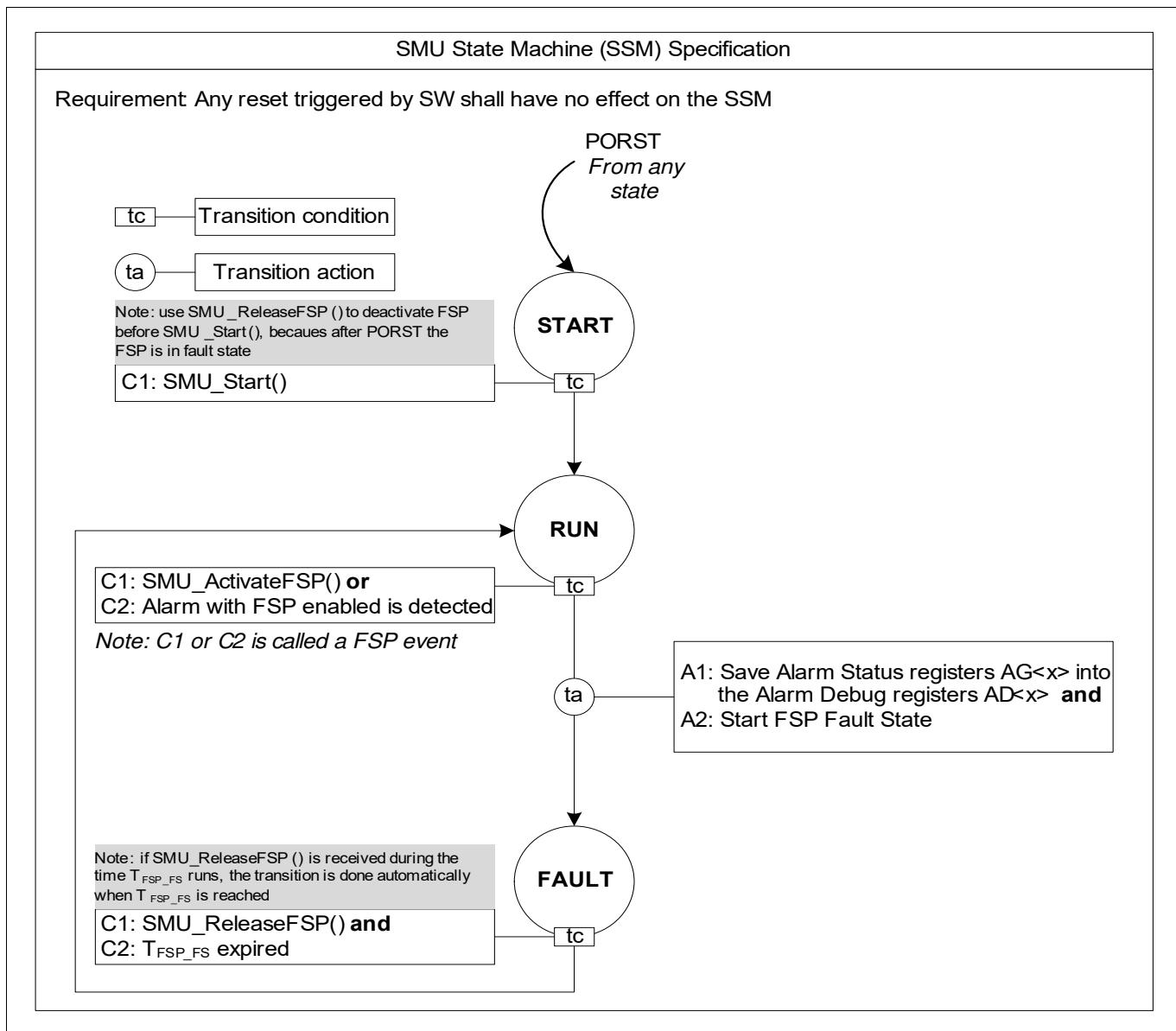
1) See also [Fault Signaling Protocol \(FSP\)](#).

2) In the START state of the SMU\_core input alarms are not processed, therefore software triggered alarms will have no effect.

## Safety Management Unit (SMU)

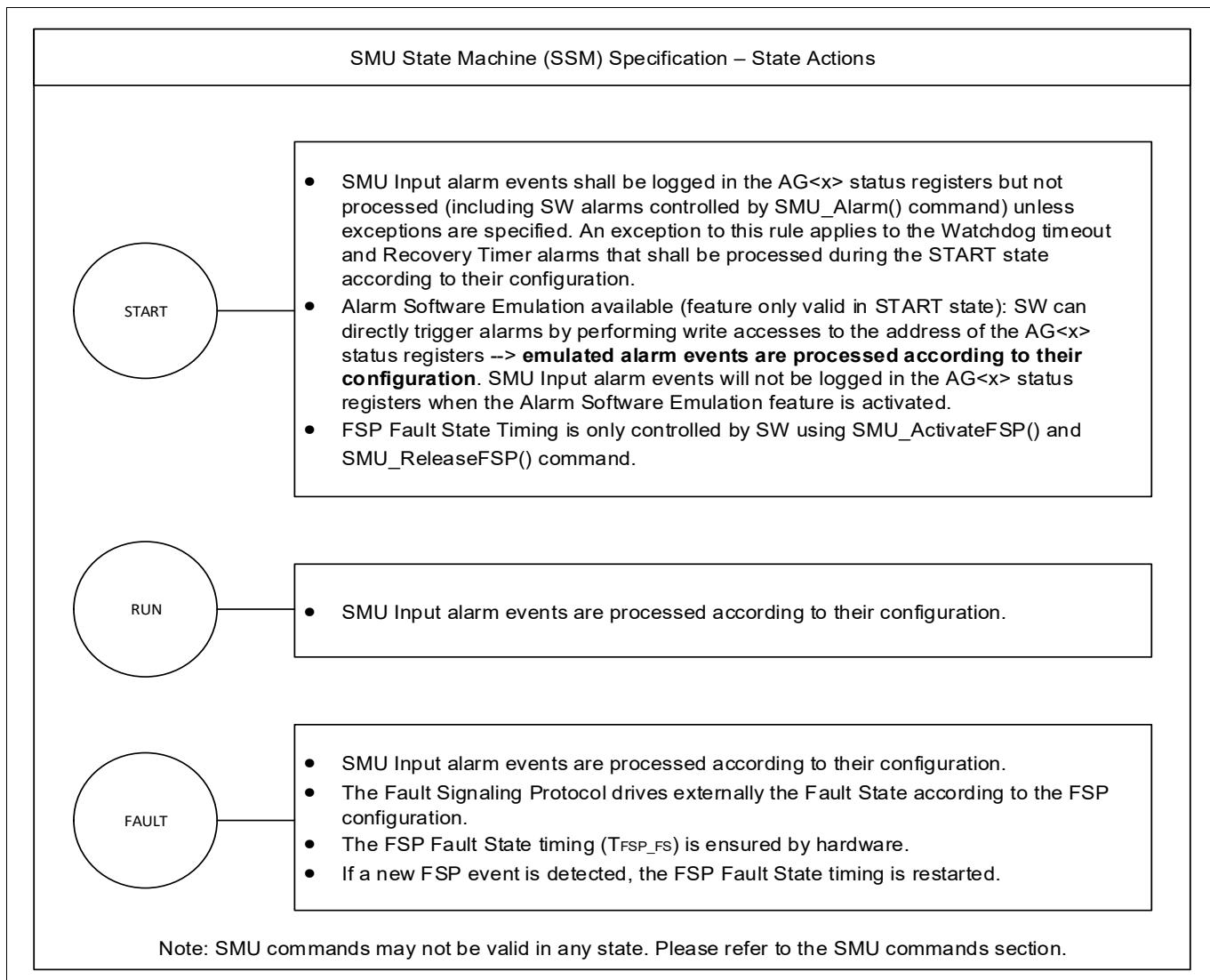
### 15.3.1.7 SMU\_core State Machine

**Figure 162** and **Figure 163** describe the behavior of the SMU\_core state machine (SSM).



**Figure 162 SMU\_core state machine (SSM): transition conditions and actions**

## Safety Management Unit (SMU)



**Figure 163 SMU\_core state machine: state actions**

### Fault Counter

The SMU implements a Fault Counter (**AFCNT**) that counts the number of transitions from the RUN state to the FAULT state. The Fault Counter register is only reset by a warm power-on-reset.

## Safety Management Unit (SMU)

### 15.3.1.8 Fault Signaling Protocol (FSP)

The Fault Signaling Protocol enables the microcontroller to report a critical situation to an external safety controller device in order to control the safe state of the safety system.

#### 15.3.1.8.1 Introduction

The fault signaling protocol is configured via the **FSP** command register. The FSP status is indicated by the FSP flag in the **STS** register. The FSP has three states:

- The power-on reset state. After warm power-on reset the SMU is disconnected from the ports (see **SMU\_core Integration Guidelines**). After warm power-on reset the SMU FSP output shall be the Fault State.
- The Fault-free State. Whenever the fault-free state is controlled by a timing, the timing will be called TFSP\_FFS and is controlled by the **FSP** register.
- The Fault State. The timing of the fault state is controlled by the **FSP** register. The minimum active fault state time is called TFSP\_FS.

The Fault-free and fault state behavior can be configured with the following protocols:

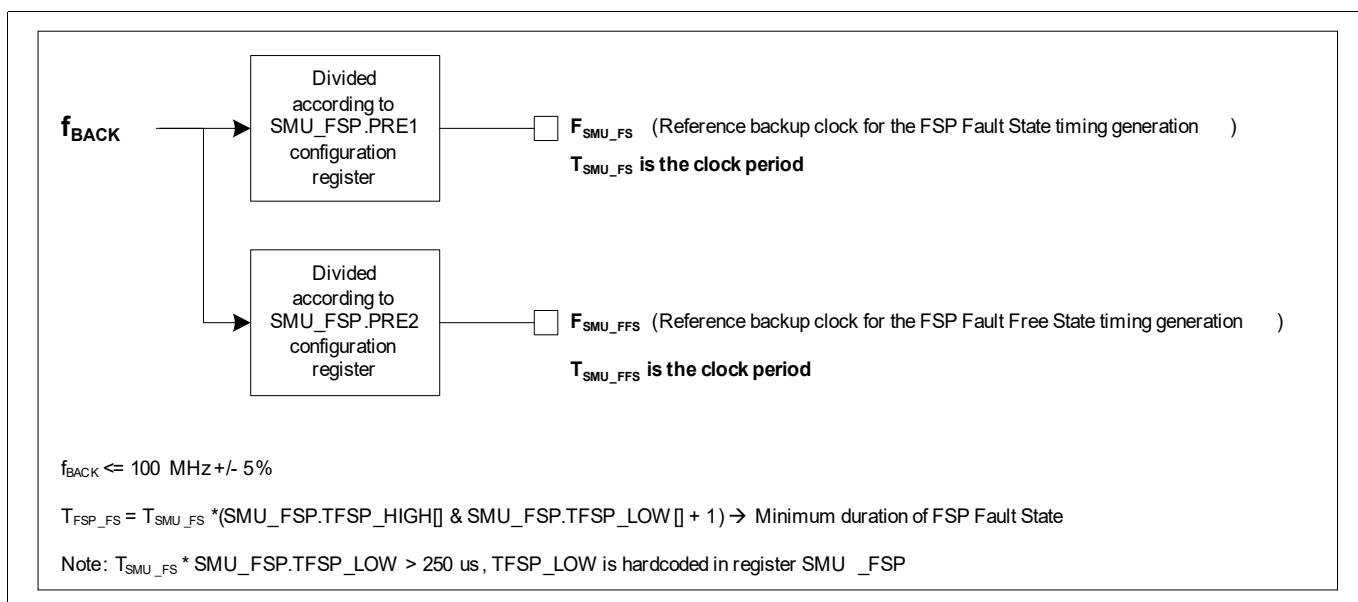
- Bi-stable protocol (default)
- Dynamic dual-rail protocol
- Time-switching protocol

The FSP can be controlled by:

- Software using the **SMU\_ActivateFSP()** and **SMU\_ReleaseFSP()** commands using the **CMD** register
- Hardware based on the **AGiFSP (i=0-11)** configuration registers.

To avoid unexpected alarms, perform the configuration of the PRE1, PRE2 or TFSP\_HIGH fields only when the SMU is not in the Fault State and FSP is in Bi-stable protocol mode (FSP.MODE = 00B). Mode switching and configuration shall not be done with the same write access to register FSP. If field FSP.PRE1 shall be written, make sure no recovery timer is running (state of recovery timer is indicated by bits RTS0 and RTS1 in the STS register).

**Figure 164** specifies the intermediate clocks to generate the TFSP\_FFS and TFSP\_FS timings.



**Figure 164 Reference clocks for FSP timings**

## Safety Management Unit (SMU)

### 15.3.1.8.2 Bi-stable fault signaling protocol

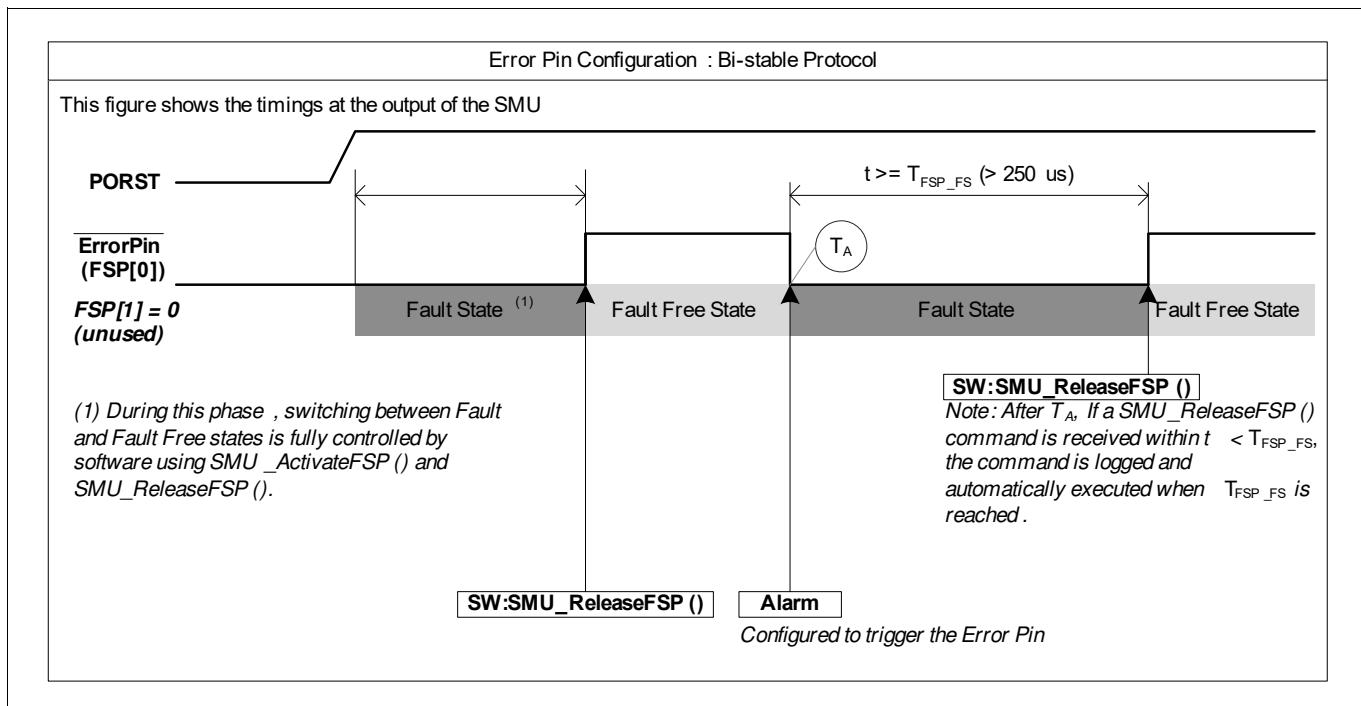


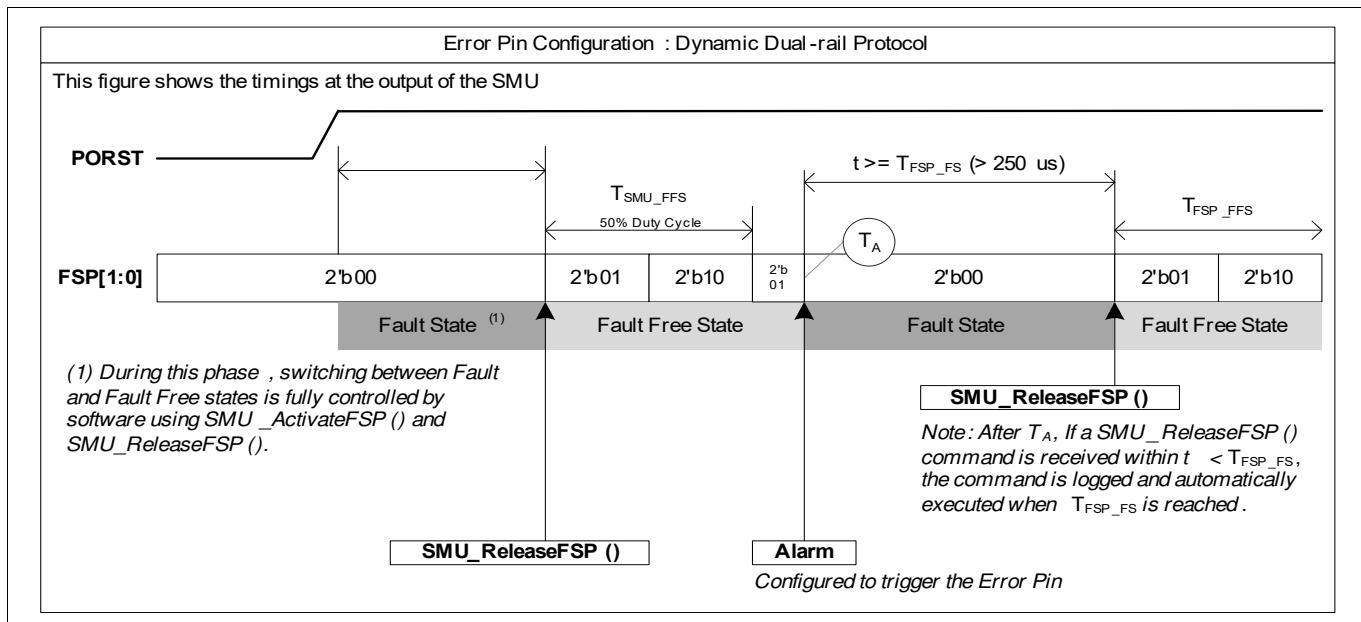
Figure 165 Bi-stable fault signaling protocol

#### Operation

- During power-on-reset  $FSP[0] = 0$  (fault state).
- After power-on reset  $FSP[0]$  stays in the fault state.
- $FSP[0]$  must be set to the fault free state per software ( $SMU\_ReleaseFSP()$ ).
- Upon detection of an alarm event configured to activate the FSP,  $FSP[0]$  goes to the fault state and remains in this state until a  $SMU\_ReleaseFSP()$  command is received and  $TFSP\_FS$  is satisfied or a Power-on Reset takes place.
- While in the Fault State, if a new alarm event configured to activate the FSP is received and the  $TFSP\_FS$  has not yet been reached, the  $TFSP\_FS$  timing shall be restarted.
- While in the Fault State, if a new alarm event configured to activate the FSP is received and the  $TFSP\_FS$  has already been reached, the  $TFSP\_FS$  timing shall be started.

## Safety Management Unit (SMU)

### 15.3.1.8.3 Timed dual rail



**Figure 166 Dynamic dual-rail fault signaling protocol**

#### Operation

Dual-rail encoding is an alternate method for encoding bits. Dual-rail codes use two signals to define a logical state.

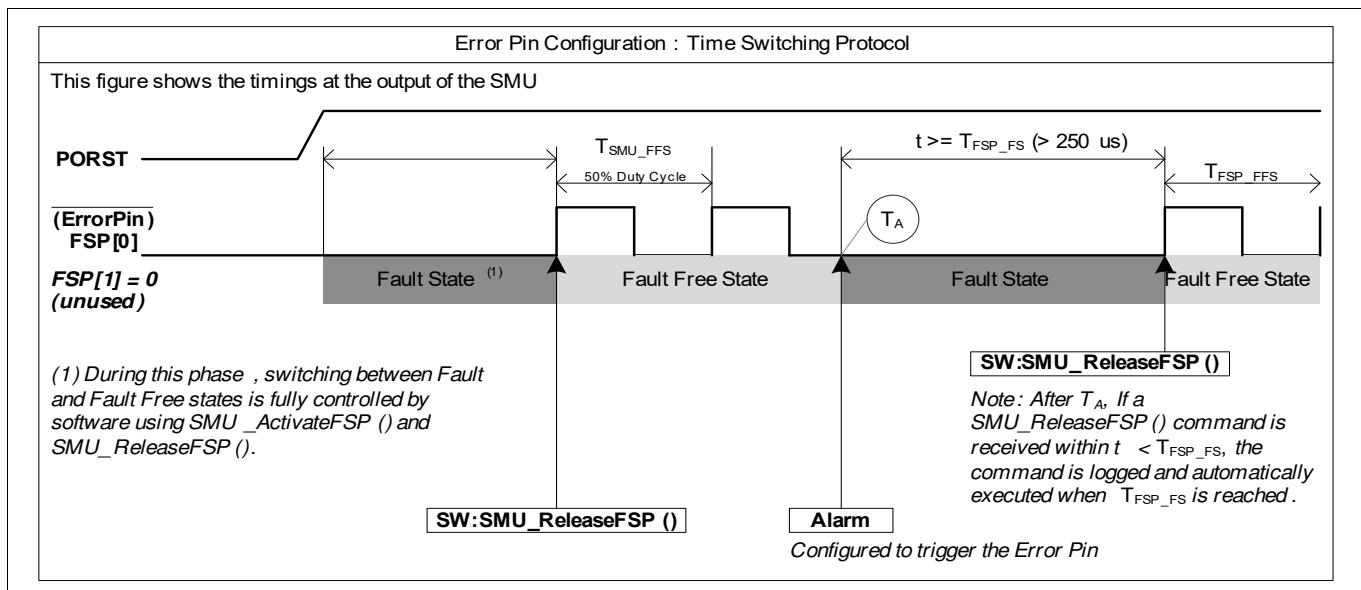
- During power-on-reset FSP[1:0] = 2'b00 (fault state)
- After power-on-reset FSP[1:0] stays in the fault state.
- FSP[1:0] must be set to the fault free state per software(SMU\_ReleaseFSP()).
- The fault free state is defined by FSP[1:0] oscillating between 2'b01 and 2'b10 with a defined frequency configured via the **FSP** register and with a duty cycle of 50% (see **Figure 166**).
- Upon detection of an alarm event configured to activate the FSP, FSP[1:0] goes immediately to the fault state and remains in this state until a SMU\_ReleaseFSP() command is received and TFSP\_FS is satisfied or a Power-on Reset takes place.

**Table 535 Dual rail coding**

Code	Description
{0,1}	Fault free state
{1,0}	Fault free state
{0,0}	Fault state
{1,1}	Fault state (encoding not used)

### 15.3.1.8.4 Time switching protocol

## Safety Management Unit (SMU)



**Figure 167 Time switching protocol**

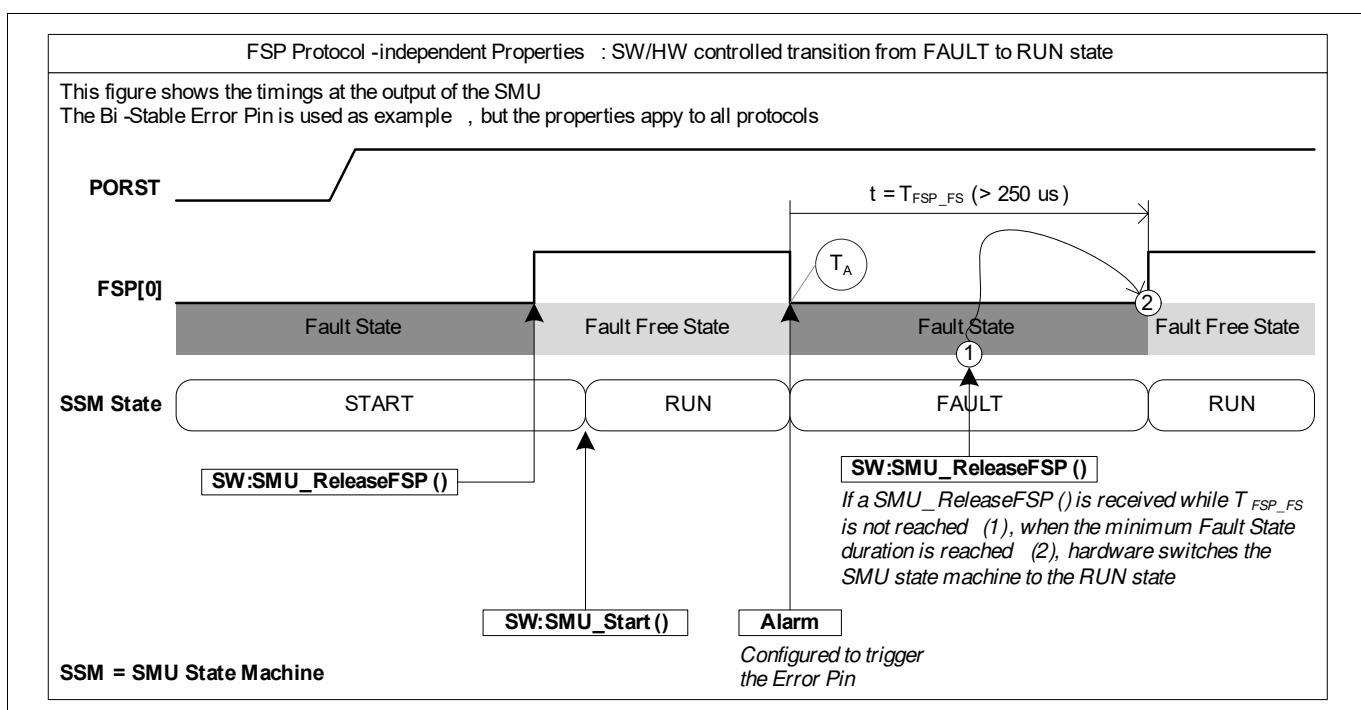
FSP[0] is toggled between logic level 0 and logic level 1 with a defined frequency. This frequency modulation protocol is violated when the SMU\_core enters the FAULT state.

- During power-on-reset FSP[0] = 0 (fault state).
- After power-on-reset FSP[0] stays in the fault state.
- FSP[0] must be set to the fault free state per software (SMU\_ReleaseFSP()).
- In the fault free state, FSP[0] oscillates between logic level 0 and logic level 1 with the frequency configured via the **FSP** register (see [Figure 167](#)).
- Upon detection of an alarm event configured to activate the FSP, FSP[0] goes immediately to the fault state and remains in this state until a SMU\_ReleaseFSP() command is received and TFSP\_FS is satisfied or a Power-on Reset takes place.

## Safety Management Unit (SMU)

### 15.3.1.8.5 FSP Fault State

When an alarm configured to activate the FSP, the SMU\_core automatically switches to the FAULT state. During this time it is also possible for the safety-related software to try to analyze the root cause (when the microcontroller is still operational) and decide about the severity of the error. As the FSP is active for at least  $T_{FSP\_FS}$ , it is ensured that the safe state of the system is entered through external mechanisms independent from the microcontroller (besides FSP itself). During the time  $T_{FSP\_FS}$  FSP is in the Fault State, software may have concluded the fault is uncritical and decides to issue a `SMU_ReleaseFSP()` command, notifying the SMU\_core that it can return to the RUN state (the run-time of the software error handler is not directly correlated with the  $T_{FSP\_FS}$  duration and in practice shall be much shorter).



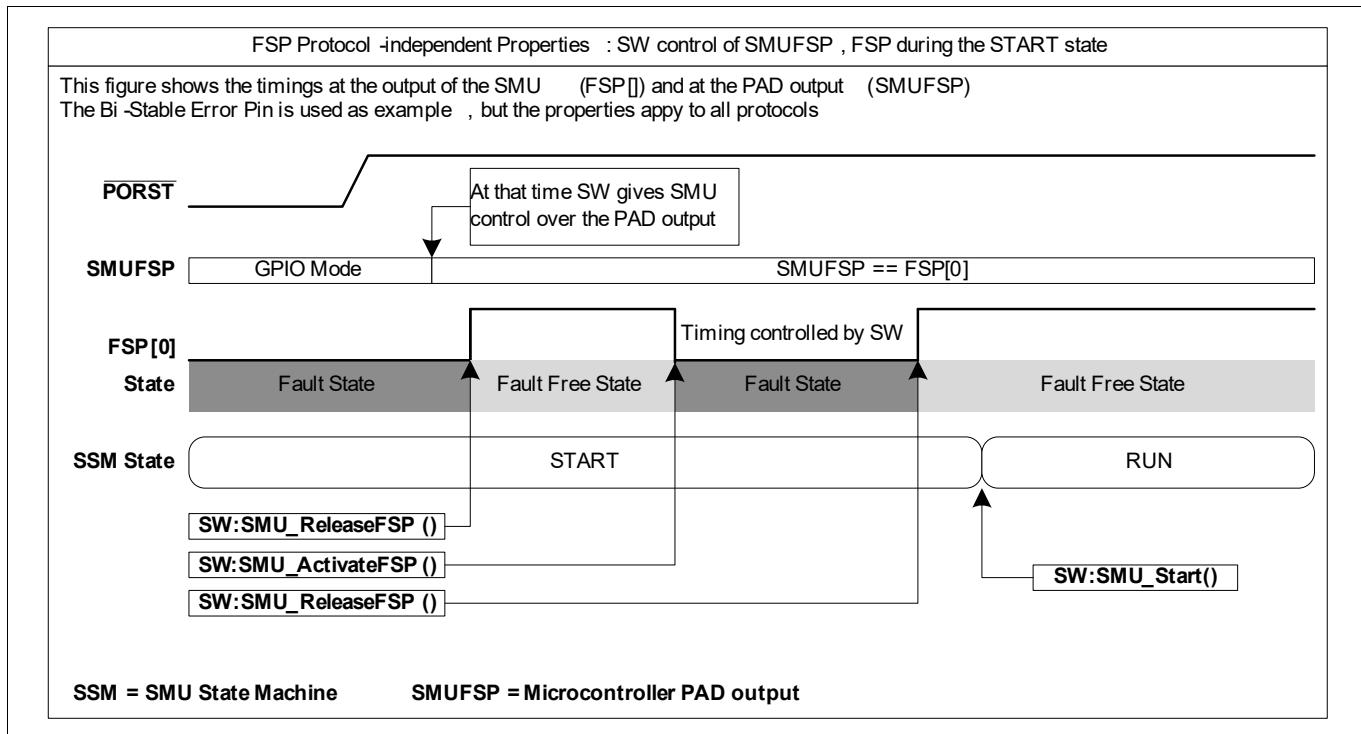
**Figure 168 FSP: Fault State to Fault Free State Transition**

This feature shall be used with caution, a microcontroller reset is highly recommended to restart the operation of the safety function when a fault reported by the SMU is assessed to be uncritical. Therefore this feature is per default disabled and shall be configured with the EFRST (Enable Fault to Run State Transition) field in the **AGC** register.

## Safety Management Unit (SMU)

### 15.3.1.8.6 FSP and SMU\_core START State

**Figure 169** shows a typical use case where the FSP transitions between the Fault State and Fault Free State are controlled by software using the SMU\_ReleaseFSP() and SMU\_ActivateFSP() commands.



**Figure 169 Software Control of FSP during SMU\_core START State**

#### Conditions of Use

- Software shall ensure that the FSP is in the Fault Free State before entering the RUN state with the SMU\_Start() command.

## Safety Management Unit (SMU)

### 15.3.1.9 OCDS Trigger Bus (OTGB) Interface

The SMU\_core concentrates all failure indicator signals (alarms) of the device. By using them as MCDS trace information (failure type) and trace control (stop trace recording), the analysis of a failure's root cause is supported. This is even the case, when the alarm handling within the system includes a PORST, since the content of the trace memory will be still valid after the PORST.

The alarms handled in the SMU\_core are very rare and sporadic. So it's acceptable that they are visible on the Trigger Bus with a small delay and not all in parallel with a cycle accurate timing resolution. The SMU Trigger Set is shown in **Table 536**. It is output on OTGB0 or OTGB1 controlled by the **OCS** register.

**Table 536 TS16\_SMU Trigger Set SMU**

Bits	Name	Description
0	AA	Any ALM bit active
1		Reserved
[3:2]	ABI	ALM Byte Index, selecting the byte within the ALM group
[7:4]	AGI	ALM Group Index
[15:8]	ALM	Selected byte of the ALM group

If no alarm is active, TS16\_SMU will be all zero. If alarms are only active within one ALM Byte, TS16\_SMU will show statically this byte. If alarms are active in two or more ALM Bytes, TS16\_SMU will change between these bytes.

Note that due to the implementation, that all the ALM Bytes are being scanned one per clock cycle, the TS16\_SMU ALM Byte information on OTGB0/1 can be delayed by up to 51 clock cycles for the second alarm. TS16\_SMU.AA will however become active (and inactive) immediately. The implementation will make sure, that the ALM Byte with the first causing alarm will be traced with MCDS before a reset as safety action clears this information.

**Note:** *In some cases, it might be possible to clear an alarm faster than it is captured on the OTGB. Therefore software needs to take care that the time from an incoming alarm until it is cleared is longer than 48 SPB clock cycles to ensure that the alarm is captured on the OTGB.*

## Safety Management Unit (SMU)

### 15.3.1.10 Register Properties

#### 15.3.1.10.1 Register Write Protection

The SMU\_core registers are write protected against illegal master accesses by the master protection mechanism implemented in the System Peripheral Bus interface logic. The registers controlling the master access protection are described in the section [System Registers description](#). In addition the SMU\_core registers can only be written if the Safety ENDINIT is enabled. Read accesses have no restriction as there is no side effect specified when reading registers. The Safety ENDINIT has the same properties as the system ENDINIT but it is generated by the safety watchdog. In order to access a SMU\_core register the software must first activate the safety watchdog via a signature protected sequence. The safety watchdog is configured to enable only safety-related software to activate the Safety ENDINIT.

In addition to the aforementioned standard features, additional mechanisms are implemented to control and protect the SMU\_core configuration. In order to configure and lock the SMU\_core configuration the following steps shall be followed:

- The SMU\_core configuration is only possible if the CFGLCK field of the **KEYS** register is set to 0xBC.
- If the PERLCK field of the **KEYS** register is set to 0xFF no further SMU configuration is possible, including the **KEYS** register itself. No SMU configuration is possible anymore until the PERLCK field of the **KEYS** register is reset to 0x00 by an application reset.

The SMU configuration registers controlled by the **KEYS** register properties are:

- **FSP**
- **AGC**
- **RTC**
- **RTAC00**
- **RTAC01**
- **RTAC10**
- **RTAC11**
- **AGiCF<sub>j</sub> (i=0-11;j=0-2)**
- **AGiFSP (i=0-11)**
- **PCTL**
- **RMCTL**

The **CMD** register is not locked as it is used for run-time hardware/software interaction, it is not a configuration register.

The SMU\_AGx registers are not locked as it is possible during run-time to clear the alarm by software.

The **OCS**, **ACCEN0** and **ACCEN1** do not belong to the SMU\_core kernel but to the standard bus interface module, therefore they can't be controlled by the **KEYS** register.

The read only registers do not need to be protected.

#### 15.3.1.10.2 Safety Flip-flops

Safety flip-flops are special flip-flops that implement an hardware mechanism capable to detect single event effects that may lead to single event upsets (bit flip). The SMU\_core configuration and control registers that shall be implemented with safety flip-flops are:

- **FSP**
- **CMD**

## Safety Management Unit (SMU)

- AGC
- RTC
- KEYS
- PCTL
- RTAC00
- RTAC01
- RTAC10
- RTAC11
- AEX
- AGiCF<sub>j</sub> (i=0-11;j=0-2)
- AGiFSP (i=0-11)

Additionally the following SMU\_core functions shall also be implemented with safety flip-flops:

- SMU\_core state machine registers
- Registers implementing the FSP function

### 15.3.2 SMU\_stby

#### 15.3.2.1 Reset Types

The SMU\_stby requires multiple reset types for its operation. The reset types are fully specified in the Power Management System. The reset types that are required by the SMU\_stby are:

- Warm Power-on Reset.
- LVD Reset.

**Table 537** specifies the scope of each reset type to the SMU\_stby functions (a function includes the control and configuration registers and the related logic).

**Table 537 Effect of Reset Types to SMU\_stby functionality**

SMU Function	Warm Power-on Reset	LVD Reset
SMU_stby Alarm Status Registers <b>Chapter 15.4.2.3</b>	Not Affected	Reset
SMU_stby Alarm Configuration Functionalities	Reset	Reset
SMU_stby BIST functionality	Reset	Reset
SMU_stby other functions	Reset	Reset

#### 15.3.2.2 Interfaces Overview

This section describes the main interface signals between the SMU\_stby and the other modules.

##### 15.3.2.2.1 Interface to the Pads (ErrorPin)

The SMU\_stby has the ability to signal an error to the external world via the FSP ErrorPin.

**Figure 170** fully specifies the data path connectivity of the SMU\_stby with the FSP ErrorPins.

## Safety Management Unit (SMU)

The ErrorPin is connected to the SMU\_stdby via two enable signals, FSP0EN and FSP1EN, and the ENPS signal. FSP0EN and FSP1EN are controlled by the bitfields **CMD\_STDBY**.FSP0EN, **CMD\_STDBY**.FSP1EN. They enable the SMU\_stdby to use the ErrorPins.

When **CMD\_STDBY**.FSP0EN and **CMD\_STDBY**.FSP1EN are set, the ENPS signal can be driven active by the SMU\_stdby which would set FSP[1..0] in high impedance state regardless of the port configuration and of SMU\_core actions.

In this case, an external pull-down device might be needed to make the high impedance state correspond to the FSP Fault State.

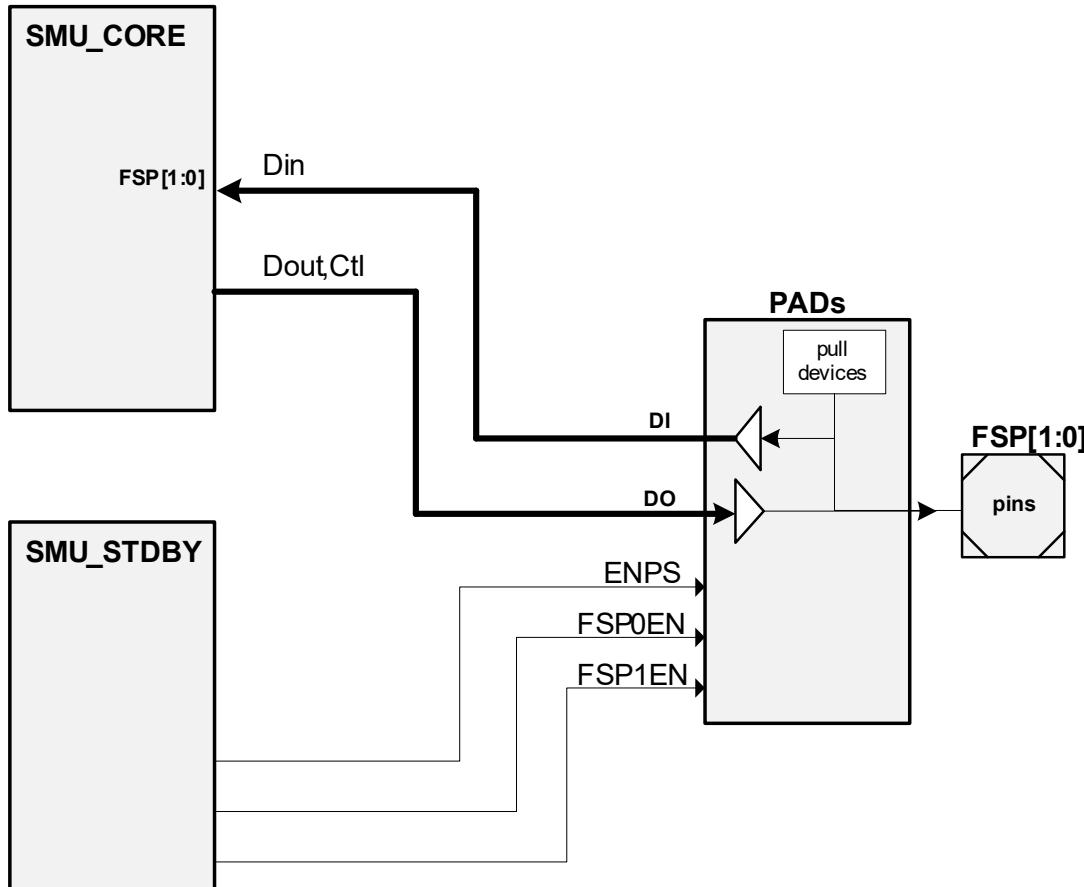


Figure 170 SMU\_stdby Data Path Interfaces with the ErrorPins

---

**Safety Management Unit (SMU)****15.3.2.3 Alarm Mapping**

Please refer to the appendix document for device specific alarm tables.

**15.3.2.3.1 SMU\_stby Internal Alarms**

The following tables describe the alarms generated by the SMU\_stby.

**SMU\_STDBY Alarm Table****Table 538 SMU\_STDBY Alarm Table**

Description	SMU Targets
SMU.SMU_stby - Safety flip-flop Uncorrectable error	ALM21[7]

---

## Safety Management Unit (SMU)

### 15.3.2.4 Alarm Handling

This section specifies the hardware and software alarm processes.

#### 15.3.2.4.1 Alarm protocol

Each safety mechanism shall interface with the SMU\_stdby using a pre-defined protocol. The protocol enables to cross clock domains in a reliable manner. The operation of the protocol has no influence to the software layers.

#### 15.3.2.4.2 Alarm Configuration

Upon reception of an alarm event the SMU\_stdby decodes the actions to be performed. The action can either be not to generate any reaction or to set FSP[1..0] in high impedance state. Both behaviors can be configured for every alarm.

The external behavior (setting of FSP[1..0] in high impedance state) is configured via the following registers:

- **AG2iFSP\_STDBY (i=0)**
- **AG2iFSP\_STDBY (i=1)**

*Note:* *In order to recognize the high impedance state of the ErrorPins as the fault state, an external pull-down device might be necessary.*

**Attention:** *When the SMU\_stdby, as a reaction to an alarm, sets the FSP[0] or/and FSP[1] in high impedance state, the STS.FSP[0] or/and STS.FSP[1] will be set to 1 if no external pull-down device is connected to the respective ErrorPin.*

## Safety Management Unit (SMU)

### 15.3.2.5 Register Properties

#### 15.3.2.5.1 Register Write Protection

The SMU\_stdby registers are write protected against illegal master accesses by the master protection mechanism implemented in the System Peripheral Bus interface logic. In addition the SMU\_stdby registers can only be written if the Safety ENDINIT is enabled. Read accesses have no restriction as there is no side effect specified when reading registers. The Safety ENDINIT has the same properties as the system ENDINIT but it is generated by the safety watchdog. In order to access a SMU\_stdby register the software must first activate the safety watchdog via a signature protected sequence. The safety watchdog is configured to enable only safety-related software to activate the Safety ENDINIT.

The read only registers do not need to be protected.

#### 15.3.2.5.2 Safety Flip-flops

Safety flip-flops are special flip-flops that implement an hardware mechanism capable to detect single event effects that may lead to single event upsets (bit flip). The SMU\_stdby configuration and control registers that shall be implemented with safety flip-flops are:

- [AG2iFSP\\_STDBY \(i=0\)](#), [AG2iFSP\\_STDBY \(i=1\)](#)
- [MONBISTCTRL](#)
- [CMD\\_STDBY](#)

#### 15.3.2.6 SMU\_stdby Built-In Self Test

The SMU\_stdby contains a built-in mechanism that enables users to test all alarm paths, alarm configurations, and alarm reactions. The [MONBISTCTRL](#) register enables the user to start the BIST of the SMU\_stdby. Results of the BIST are available in the [MONBISTSTAT](#) register. Please refer to the Power Management Unit Chapter for more details about the SMU\_stdby BIST operations.

### 15.3.3 Interdependency Between SMU\_core and SMU\_stdby

The SMU\_core and SMU\_stdby are designed to function together. The SMU\_stdby monitors the SMU\_core and can, via the ErrorPins, notify an external device of a fault in the SMU\_core.

All alarms (except for PLLx/fSPB and SMU\_core alive alarms) that are processed by the SMU\_stdby are also forwarded to the SMU\_core and processed whether or not the SMU\_stdby is enabled. Also, the SMU\_stdby alarm status registers are accessible even when the SMU\_stdby is disabled. SMU\_core Alive alarm can be monitored when SMU\_stdby is disabled. Thus, a user could decide to run an application with both SMU\_core and SMU\_stdby enabled or with the SMU\_core enabled and the SMU\_stdby disabled.

However, the Fault Signaling Protocol is generated by the SMU\_core. As a consequence, if the SMU\_core is not enabled and in run state, the FSP ErrorPins are in their default state, which is the FSP fault state. In such a configuration, the reaction of the SMU\_stdby to any alarm (setting the ErrorPins in fault state) will not be noticeable.

Thus, the SMU\_stdby cannot be used to react to alarms when the SMU\_core is not enabled and in run state.

PMS Alarms occurring during warm PORST will not be latched neither in the SMU\_core nor in the SMU\_stdby Alarm Status Registers.

## 15.4 Registers

This section describes the SMU\_core module and SMU\_stdby module registers.

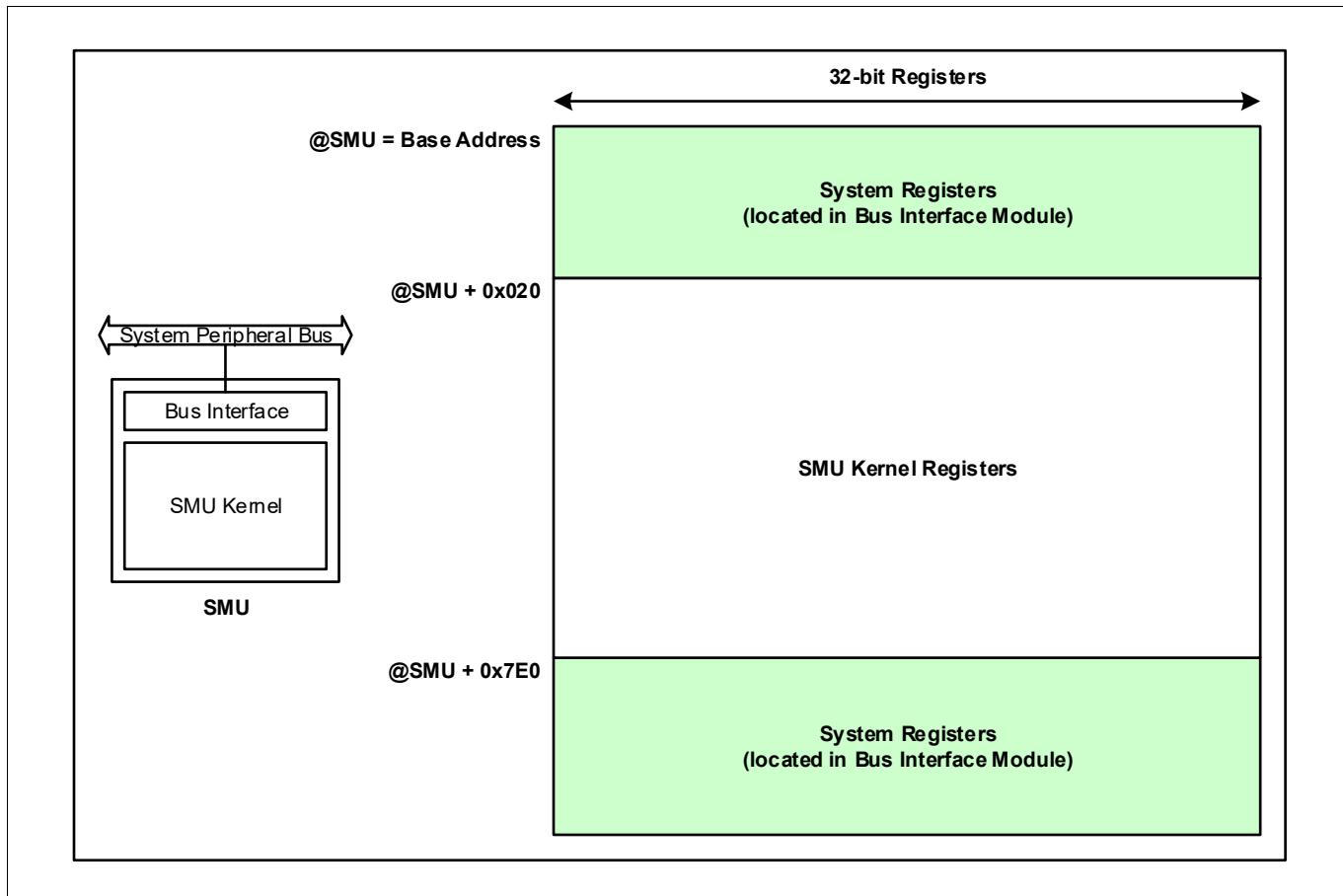
## Safety Management Unit (SMU)

### 15.4.1 SMU\_core Module Registers

**Figure 171** shows the SMU\_core module register map.

**Table 539** shows the SMU\_core Address Space

**Table 540** lists all registers implemented in the SMU\_core.



**Figure 171** SMU\_core

**Table 539 Register Address Space - SMU**

Module	Base Address	End Address	Note
SMU	F0036800 <sub>H</sub>	F0036FFF <sub>H</sub>	FPI slave interface

**Table 540 Register Overview - SMU (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
CLC	Clock Control Register	000 <sub>H</sub>	U,SV	SV,P	Application Reset	<b>40</b>
ID	Module Identification Register	008 <sub>H</sub>	U,SV	BE	Application Reset	<b>41</b>
CMD	Command Register	020 <sub>H</sub>	U,SV	SV,P,32	Application Reset	<b>44</b>

**Safety Management Unit (SMU)****Table 540 Register Overview - SMU (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
STS	Status Register	024 <sub>H</sub>	U,SV	SV,P,32	Application Reset	<a href="#">44</a>
FSP	Fault Signaling Protocol	028 <sub>H</sub>	U,SV	SV,P,SE,32	PowerOn Reset	<a href="#">46</a>
AGC	Alarm Global Configuration	02C <sub>H</sub>	U,SV	SV,P,SE,32	Application Reset	<a href="#">47</a>
RTC	Recovery Timer Configuration	030 <sub>H</sub>	U,SV	SV,P,SE,32	Application Reset	<a href="#">49</a>
KEYS	Key Register	034 <sub>H</sub>	U,SV	SV,P,SE,32	Application Reset	<a href="#">50</a>
DBG	Debug Register	038 <sub>H</sub>	U,SV	BE	PowerOn Reset	<a href="#">50</a>
PCTL	Port Control	03C <sub>H</sub>	U,SV	SV,P,SE,32	PowerOn Reset	<a href="#">51</a>
AFCNT	Alarm and Fault Counter	040 <sub>H</sub>	U,SV	BE	PowerOn Reset	<a href="#">52</a>
RTAC00	Recovery Timer 0 Alarm Configuration 0	060 <sub>H</sub>	U,SV	SV,P,SE,32	Application Reset	<a href="#">53</a>
RTAC01	Recovery Timer 0 Alarm Configuration 1	064 <sub>H</sub>	U,SV	SV,P,SE,32	Application Reset	<a href="#">54</a>
RTAC10	Recovery Timer 1 Alarm Configuration 0	068 <sub>H</sub>	U,SV	SV,P,SE,32	Application Reset	<a href="#">54</a>
RTAC11	Recovery Timer 1 Alarm Configuration 1	06C <sub>H</sub>	U,SV	SV,P,SE,32	Application Reset	<a href="#">55</a>
AEX	Alarm Executed Status Register	070 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">56</a>
AEXCLR	Alarm Executed Status Clear Register	074 <sub>H</sub>	U,SV	SV,P,SE,32	Application Reset	<a href="#">60</a>
AGICFj	Alarm Configuration Register	100 <sub>H</sub> +i*1 2+j*4	U,SV	SV,P,SE,32	See page <a href="#">64</a>	<a href="#">64</a>
AGIFSP	SMU_core FSP Configuration Register	190 <sub>H</sub> +i*4	U,SV	SV,P,SE,32	See page <a href="#">64</a>	<a href="#">64</a>
AGi	Alarm Status Register	1C0 <sub>H</sub> +i*4	U,SV	SV,P,SE,32	Application Reset	<a href="#">66</a>
ADI	Alarm Debug Register	200 <sub>H</sub> +i*4	U,SV	BE	PowerOn Reset	<a href="#">66</a>
RMCTL	Register Monitor Control	300 <sub>H</sub>	U,SV	SV,P,SE,32	Application Reset	<a href="#">67</a>
RMEF	Register Monitor Error Flags	304 <sub>H</sub>	U,SV	SV,P,SE,32	Application Reset	<a href="#">67</a>
RMSTS	Register Monitor Self Test Status	308 <sub>H</sub>	U,SV	SV,P,SE,32	Application Reset	<a href="#">68</a>
OCS	OCDS Control and Status	7E8 <sub>H</sub>	U,SV	SV,P,OEN	Debug Reset	<a href="#">41</a>

**Safety Management Unit (SMU)****Table 540 Register Overview - SMU (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
ACCEN1	SMU_core Access Enable Register 1	7F8 <sub>H</sub>	U,SV	BE	Application Reset	<b>43</b>
ACCENO	SMU_core Access Enable Register 0	7FC <sub>H</sub>	U,SV	SV,SE	Application Reset	<b>42</b>

## **Safety Management Unit (SMU)**

#### **15.4.1.1 System Registers description**

## Clock Control Register

The Clock Control Register allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The SMU\_core shall be enabled per default.

**Note:** The other features controlled by the CLC register are not supported by the SMU\_core.

**CLC**  
**Clock Control Register** (000<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>

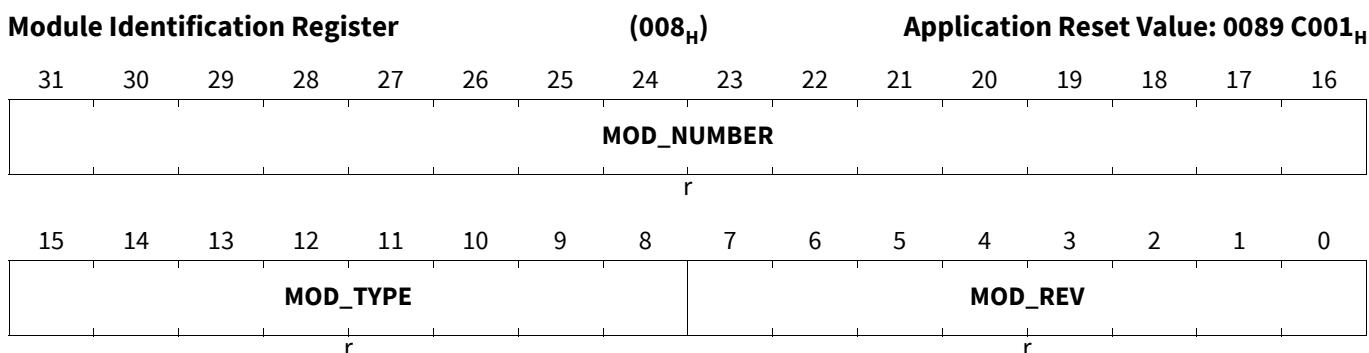
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															
												EDIS	0	DISS	DISR
												rw	r	rh	rw

Field	Bits	Type	Description
<b>DISR</b>	0	rw	<p><b>Module Disable Request Bit</b></p> <p>Used for enable/disable control of the module.</p> <p>0<sub>B</sub> Module disable is not requested 1<sub>B</sub> Module disable is requested</p>
<b>DISS</b>	1	rh	<p><b>Module Disable Status Bit</b></p> <p>Bit indicates the current status of the module.</p> <p>0<sub>B</sub> Module is enabled 1<sub>B</sub> Module is disabled</p>
<b>EDIS</b>	3	rw	<p><b>Sleep Mode Enable Control</b></p> <p>Used to control module's sleep mode.</p> <p>Sleep Mode is not supported by the safety applications. During the process of entering and resuming from sleep mode, the intended processing of alarm events is not guaranteed.</p>
<b>0</b>	2, 31:4	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

## Safety Management Unit (SMU)

### Module Identification Register

ID



Field	Bits	Type	Description
<b>MOD_REV</b>	7:0	r	<b>Module Revision Number</b> This bit field defines the module revision number. The value of a module revision starts with 01H (first revision).
<b>MOD_TYPE</b>	15:8	r	<b>Module Type</b> The bit field is set to C0H which defines the module as a 32-bit module.
<b>MOD_NUMBE R</b>	31:16	r	<b>Module Number Value</b> This bit field defines a module identification number. The value for the SMU module is 0089H.

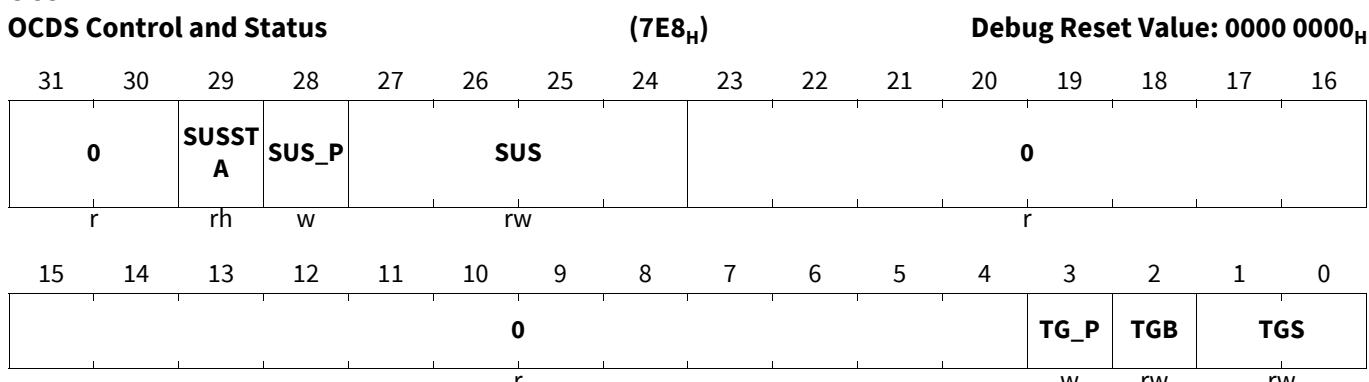
### OCDS Control and Status

SMU OCDS Control Register. This register is implemented in the BPI.

The OCDS Control and Status (OCS) register is reset by Debug Reset. The OCS register includes the module related control bits for the OCDS Trigger Bus(OTGB).

The register can only be written and the OCS control register bits are only effective while the OCDS is enabled (OCDS enable = '1'). While OCDS is not enabled, OCS reset values/modes are effective.

OCS



**Safety Management Unit (SMU)**

Field	Bits	Type	Description
<b>TGS</b>	1:0	rw	<b>Trigger Set for OTGB0/1</b> 00 <sub>B</sub> No Trigger Set output 01 <sub>B</sub> TS16_SMU <b>others</b> , reserved
<b>TGB</b>	2	rw	<b>OTGB0/1 Bus Select</b> 0 <sub>B</sub> Trigger Set is output on OTGB0 1 <sub>B</sub> Trigger Set is output on OTGB1
<b>TG_P</b>	3	w	<b>TGS, TGB Write Protection</b> TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
<b>SUS</b>	27:24	rw	<b>OCDS Suspend Control</b> Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 <sub>H</sub> Will not suspend 1 <sub>H</sub> Hard suspend. Clock is switched off immediately. 2 <sub>H</sub> Soft suspend. <b>others</b> , reserved
<b>SUS_P</b>	28	w	<b>SUS Write Protection</b> SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
<b>SUSSTA</b>	29	rh	<b>Suspend State</b> Read as 0; must be written with 0.
<b>0</b>	23:4, 31:30	r	<b>Reserved</b> Read as 0; must be written with 0.

**SMU\_core Access Enable Register 0**

The Access Enable Register 0 controls write access for transactions with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI\_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

**Note:** The mapping with the on-chip master-capable modules is described in the bus chapters of the microcontroller.

**ACCEN0****SMU\_core Access Enable Register 0**      (7FC<sub>H</sub>)      Application Reset Value: FFFF FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN31</b>	<b>EN30</b>	<b>EN29</b>	<b>EN28</b>	<b>EN27</b>	<b>EN26</b>	<b>EN25</b>	<b>EN24</b>	<b>EN23</b>	<b>EN22</b>	<b>EN21</b>	<b>EN20</b>	<b>EN19</b>	<b>EN18</b>	<b>EN17</b>	<b>EN16</b>
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EN15</b>	<b>EN14</b>	<b>EN13</b>	<b>EN12</b>	<b>EN11</b>	<b>EN10</b>	<b>EN9</b>	<b>EN8</b>	<b>EN7</b>	<b>EN6</b>	<b>EN5</b>	<b>EN4</b>	<b>EN3</b>	<b>EN2</b>	<b>EN1</b>	<b>EN0</b>
rw															

## Safety Management Unit (SMU)

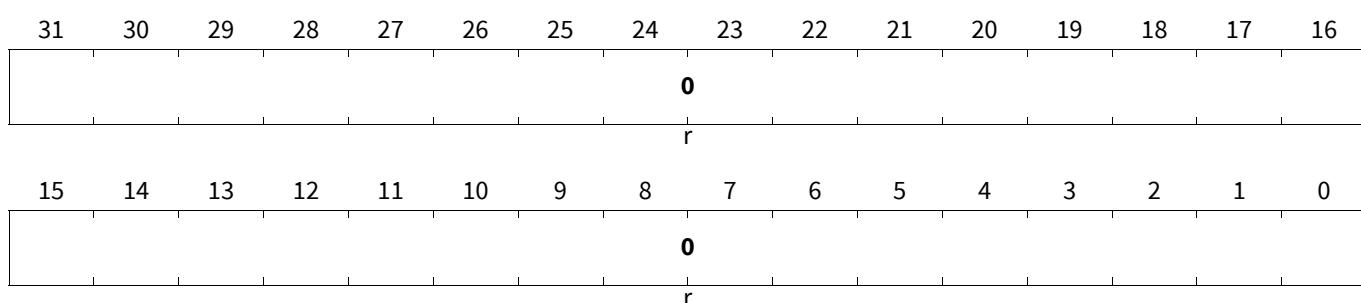
Field	Bits	Type	Description
<b>ENy (y=0-31)</b>	y	rw	<b>Access Enable for Master TAG ID y</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID y 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

### SMU\_core Access Enable Register 1

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <>> master peripheral mapping). The BPI\_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

#### ACCEN1

**SMU\_core Access Enable Register 1** **Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>0</b>	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

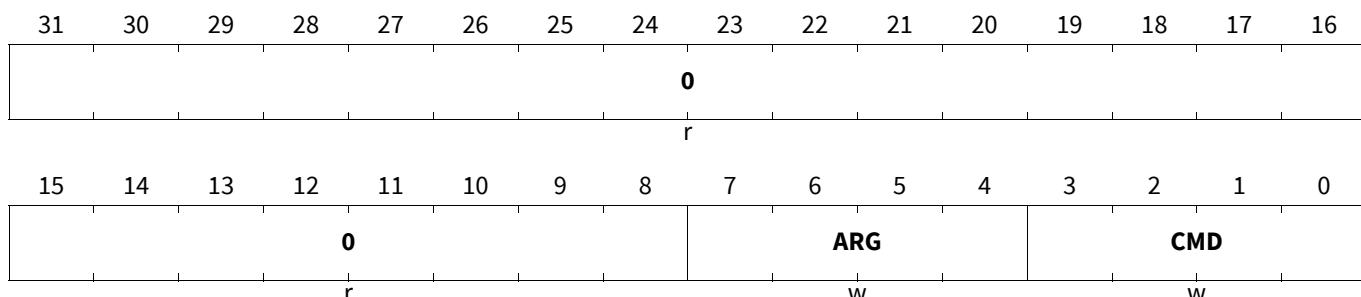
## Safety Management Unit (SMU)

### 15.4.1.2 SMU\_core Configuration Registers

#### Command Register

CMD

#### Command Register

(020<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

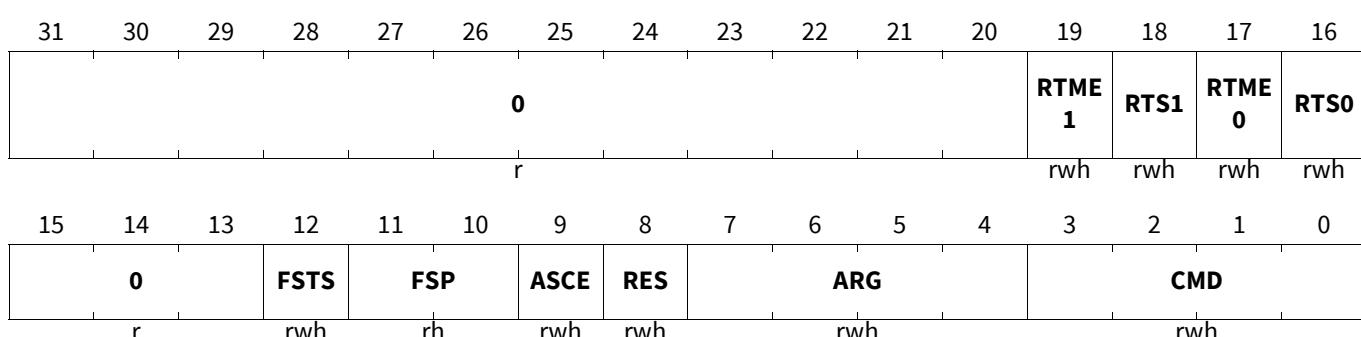
Field	Bits	Type	Description
CMD	3:0	w	<b>Implements the SMU_core Command Interface.</b> See <a href="#">Table 533 “SMU_core Commands” on Page 20</a> for the command encoding. Read as 0.
ARG	7:4	w	<b>Implements the SMU_core Command Interface.</b> Argument to be used with the command. See <a href="#">Table 533 “SMU_core Commands” on Page 20</a> for the argument encoding. Read as 0.
0	31:8	r	<b>Reserved</b> Read as 0; should be written with 0

#### Status Register

Note: A write to this register (regardless of the data written) clears all the fields with the rwh type. If on the same cycle a software write event and hardware event is detected, the hardware action wins.

STS

#### Status Register

(024<sub>H</sub>)Application Reset Value: 0000 0X00<sub>H</sub>

**Safety Management Unit (SMU)**

Field	Bits	Type	Description
<b>CMD</b>	3:0	rwh	<b>Last command received</b> Same encoding as CMD field of <a href="#">CMD register</a>
<b>ARG</b>	7:4	rwh	<b>Last command argument received</b> Same encoding as ARG field of <a href="#">CMD register</a>
<b>RES</b>	8	rwh	<b>Result of last received command</b> $0_B$ Command was successful $1_B$ Command failed
<b>ASCE</b>	9	rwh	<b>Alarm Status Clear Enable</b> This bit controls if a status flag set in an AG<x> register upon detection of an alarm event can be cleared by software or not. When ASCE is enabled software shall write a 1 to the bit position in AG<x> to clear the bit (W1C). When a W1C action takes place the ASCE bit is automatically cleared to 0 by hardware and software shall set the ASCE bit again by using the SMU_ASCE() command. $0_B$ Alarm status bits AG<x> can not be cleared $1_B$ Alarm status bits AG<x> can be cleared.
<b>FSP</b>	11:10	rh	<b>Fault Signaling Protocol status</b> FSP[0] = FSP_STS[0] input signal FSP[1] = FSP_STS[1] input signal This field is updated by hardware every clock cycle, therefore a software clear on write is not meaningful for this field.  Note: When the FSP[0] and/or FSP[1] is set in fault state by the SMU_stdby this bitfield does not reflect the actual state of the ErrorPins. Indeed, the SMU_stdby sets the ErrorPins in high impedance state to indicate the presence of a fault.
<b>FSTS</b>	12	rwh	<b>Fault State Timing Status</b> This bit indicates if the minimum timing duration of the FSP fault state has been reached or not. The bit is cleared by hardware when the fault state is entered. $0_B$ Minimum timing duration not reached $1_B$ Minimum timing duration reached
<b>RTS0</b>	16	rwh	<b>Recovery Timer 0 Status</b> See " <a href="#">Recovery Timer" on Page 18</a> for the usage of this field. $0_B$ Recovery Timer not running $1_B$ Recovery Timer running
<b>RTMEO</b>	17	rwh	<b>Recovery Timer 0 Missed Event</b> See " <a href="#">Recovery Timer" on Page 18</a> for the usage of this field. $0_B$ Recovery Timer event not detected $1_B$ Recovery Timer event detected
<b>RTS1</b>	18	rwh	<b>Recovery Timer 1 Status</b> See " <a href="#">Recovery Timer" on Page 18</a> for the usage of this field. $0_B$ Recovery Timer not running $1_B$ Recovery Timer running

## Safety Management Unit (SMU)

Field	Bits	Type	Description
<b>RTME1</b>	19	rwh	<b>Recovery Timer 1 Missed Event</b> See “ <b>Recovery Timer</b> ” on Page 18 for the usage of this field. $0_B$ Recovery Timer event not detected $1_B$ Recovery Timer event detected
<b>0</b>	15:13, 31:20	r	<b>Reserved</b> Read as 0; should be written with 0.

### Fault Signaling Protocol

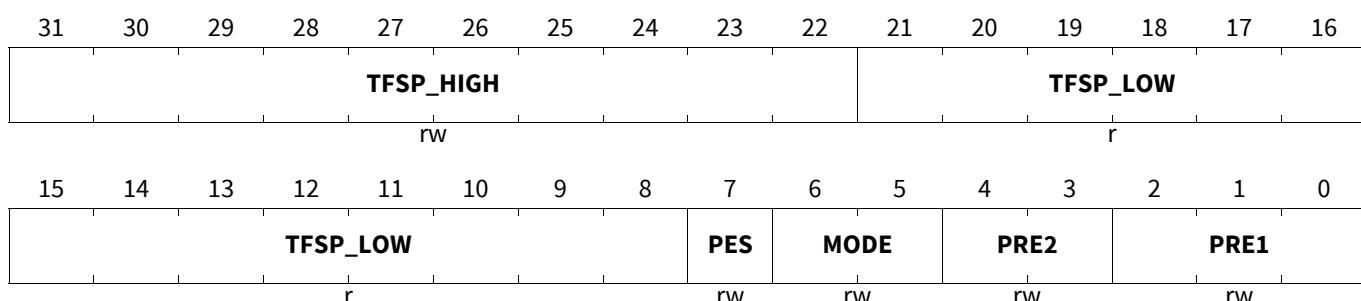
This register controls the timing of the fault signaling protocol.

#### FSP

#### Fault Signaling Protocol

(028<sub>H</sub>)

PowerOn Reset Value: 003F FF00<sub>H</sub>



Field	Bits	Type	Description
<b>PRE1</b>	2:0	rw	<b>Prescaler1</b> Dividing factor to apply to the reference clock fBACK. It is assumed that the maximal value for fBACK is 100 MHz with a precision of 5%. The divided clock is used as reference to generate the timing of the fault signaling protocol fault state.  <i>Note:</i> <i>It is only allowed to write PRE1 when the SMU is not in the Fault State and FSP is in Bi-stable protocol mode. Also, it is not allowed to write to the PRE1 when at least one recovery timer is running.</i>  The frequency of the divided clock (called FSMU_FS) is defined as follows: $000_B$ reference clock frequency divided by 2 $001_B$ reference clock frequency divided by 4 $010_B$ reference clock frequency divided by 8 $011_B$ reference clock frequency divided by 16 $100_B$ reference clock frequency divided by 32 $101_B$ reference clock frequency divided by 64 $110_B$ reference clock frequency divided by 128 $111_B$ reference clock frequency divided by 256

## Safety Management Unit (SMU)

Field	Bits	Type	Description
<b>PRE2</b>	4:3	rw	<p><b>Prescaler2</b></p> <p>Dividing factor to apply to the reference clock fBACK in order to generate the timing of the fault free state for the dynamic dual rail and time switching modes of the fault signaling protocol.</p> <p><i>Note:</i> <i>It is only allowed to write PPRE2 when the SMU is not in the Fault State and FSP is in Bi-stable protocol mode.</i></p> <p>The frequency of the divided clock (called FSMU_FFS) is defined as follows:</p> <ul style="list-style-type: none"> <li><math>00_B</math> reference clock frequency divided by 512</li> <li><math>01_B</math> reference clock frequency divided by 1024</li> <li><math>10_B</math> reference clock frequency divided by 2048</li> <li><math>11_B</math> reference clock frequency divided by 4096</li> </ul>
<b>MODE</b>	6:5	rw	<p><b>Fault Signaling Protocol configuration</b></p> <ul style="list-style-type: none"> <li><math>00_B</math> Bi-stable protocol</li> <li><math>01_B</math> Dual Rail protocol</li> <li><math>10_B</math> Time switching protocol</li> <li><math>11_B</math> Reserved</li> </ul>
<b>PES</b>	7	rw	<p><b>Port Emergency Stop (PES)</b></p> <p>When this bit is set a Port Emergency Stop is automatically requested when an alarm event configured to start the Fault Signaling Protocol is detected.</p> <ul style="list-style-type: none"> <li><math>0_B</math> Port Emergency Stop disabled</li> <li><math>1_B</math> Port Emergency Stop enabled</li> </ul>
<b>TFSP_LOW</b>	21:8	r	<p><b>Specifies the FSP fault state duration</b></p> <p>TFSP_FS= TFSP_HIGH &amp; TPSP_LOW. TFSP_LOW shall be specified as a number of FSMU_FS ticks. TFSP_LOW is defined so that the minimum duration is greater than 250 us. It can not be changed by software.</p>
<b>TFSP_HIGH</b>	31:22	rw	<p><b>Specifies the FSP fault state duration</b></p> <p>TFSP_FS= TFSP_HIGH &amp; TPSP_LOW. TFSP_HIGH shall be specified as a number of FSMU_FS ticks. TFSP_HIGH and PRE1 shall enable to configure a fault state duration of 500 ms.</p> <p><i>Note:</i> <i>It is only allowed to write TFSP_HIGH when the SMU is not in the Fault State and FSP is in Bi-stable protocol mode.</i></p>

### Alarm Global Configuration

This register controls some properties related to the behavior of the SMU\_core to alarm.

## Safety Management Unit (SMU)

### AGC

#### Alarm Global Configuration

(02C<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	EFRST		PES				0			RCS					
r	rw		rw				r			rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		0			IGCS2		0		IGCS1		0		IGCS0		
		r			rw		r		rw		r		rw		

Field	Bits	Type	Description
IGCS0	2:0	rw	<b>Interrupt Generation Configuration Set 0</b> Defines the output value of the interrupt request vector when the alarm configuration flag selects the interrupt configuration set 0. Enables to issue an interrupt request to several CPUs: see “ <a href="#">Interfaces to the Interrupt Router” on Page 7.</a>
IGCS1	6:4	rw	<b>Interrupt Generation Configuration Set 1</b> Defines the output value of the interrupt request vector when the alarm configuration flag selects the interrupt configuration set 1. Enables to issue an interrupt request to several CPUs: see “ <a href="#">Interfaces to the Interrupt Router” on Page 7.</a>
IGCS2	10:8	rw	<b>Interrupt Generation Configuration Set 2</b> Defines the output value of the interrupt request vector when the alarm configuration flag selects the interrupt configuration set 2. Enables to issue an interrupt request to several CPUs: see “ <a href="#">Interfaces to the Interrupt Router” on Page 7.</a>
RCS	21:16	rw	<b>CPU Reset Configuration Set</b> Defines the output value of the CPU reset request vector when the alarm configuration flag selects the CPU Reset Configuration Set. Enables to issue an reset request to several CPUs if required. More complex reset scenarios can be handled by using software interrupts. Setting the bit n to 1 enables issuing a reset request to CPUn.
PES	28:24	rw	<b>Port Emergency Stop</b> This field enables control of the Port Emergency Stop (PES) feature independently for each internal action. When an action is triggered and if the corresponding bit (as defined below) is set, the hardware triggers automatically a port emergency stop request. Each bit of PES is allocated to an action as follows: 01 <sub>H</sub> SMU_IGCS0 activates PES 02 <sub>H</sub> SMU_IGCS1 activates PES 04 <sub>H</sub> SMU_IGCS2 activates PES 08 <sub>H</sub> SMU_NMI activates PES 10 <sub>H</sub> SMU_CPU_RESET activates PES

## Safety Management Unit (SMU)

Field	Bits	Type	Description
<b>EFRST</b>	29	rw	<b>Enable FAULT to RUN State Transition</b> See “ <b>FSP Fault State</b> ” on Page 28 chapter for the usage of this field. 0 <sub>B</sub> FAULT to RUN State Transition disabled 1 <sub>B</sub> FAULT to RUN State Transition enabled
0	3, 7, 15:11, 23:22, 31:30	r	<b>Reserved</b> Read as 0; should be written with 0

### Recovery Timer Configuration

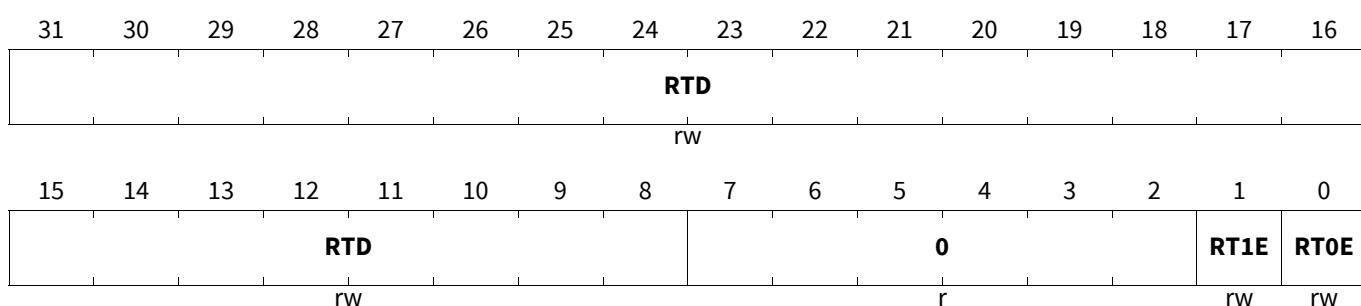
This register controls the timing duration of the recovery timer.

#### RTC

#### Recovery Timer Configuration

(030<sub>H</sub>)

Application Reset Value: 003F FF03<sub>H</sub>



Field	Bits	Type	Description
<b>RT0E</b>	0	rw	<b>RT0 Enable Bit</b> 0 <sub>B</sub> Recovery Timer 0 is disabled 1 <sub>B</sub> Recovery Timer 0 is enabled
<b>RT1E</b>	1	rw	<b>RT1 Enable Bit</b> 0 <sub>B</sub> Recovery Timer 1 is disabled 1 <sub>B</sub> Recovery Timer 1 is enabled
<b>RTD</b>	31:8	rw	<b>Recovery Timer Duration</b> This field specifies the maximum duration of the recovery timer. When the timer counter reaches the programmed value, the internal alarm rt_timeout is issued. The timer is stopped by a SMU_RTStop() command before the recovery timer. RTD shall be specified as a number of the FSMU_FS clock ticks. <i>Note:</i> It is not allowed to write to the RTD when at least one recovery timer is running.
0	7:2	r	<b>Reserved</b> Read as 0; should be written with 0

## Safety Management Unit (SMU)

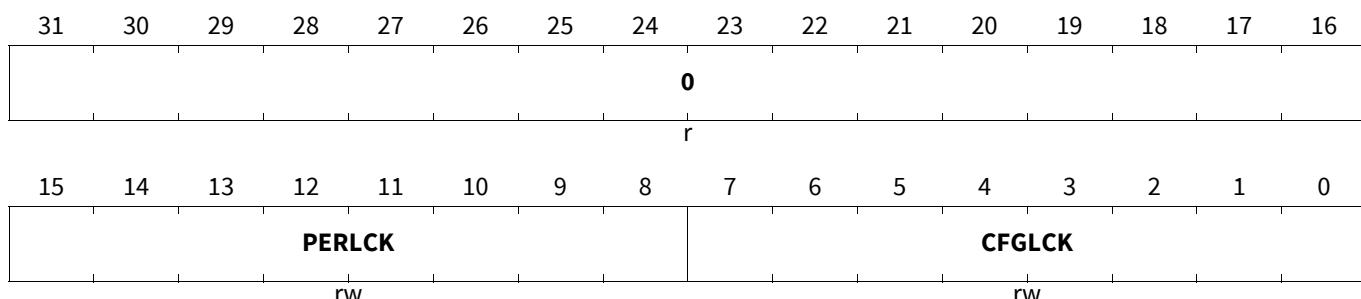
### Key Register

#### KEYS

##### Key Register

(034<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
CFGLOCK	7:0	rw	<b>Configuration Lock</b> The SMU_core configuration is only possible if this field is set to 0xBC. Refer to “ <a href="#">Register Properties</a> on Page 31” for the list of registers controlled by this field.
PERLCK	15:8	rw	<b>Permanent Lock</b> If this field is set to 0xFF, no further configuration of the SMU_core is possible. Refer to “ <a href="#">Register Properties</a> on Page 31” for the list of registers controlled by this field.
0	31:16	r	<b>Reserved</b> Read as 0; shall be written with 0

### Debug Register

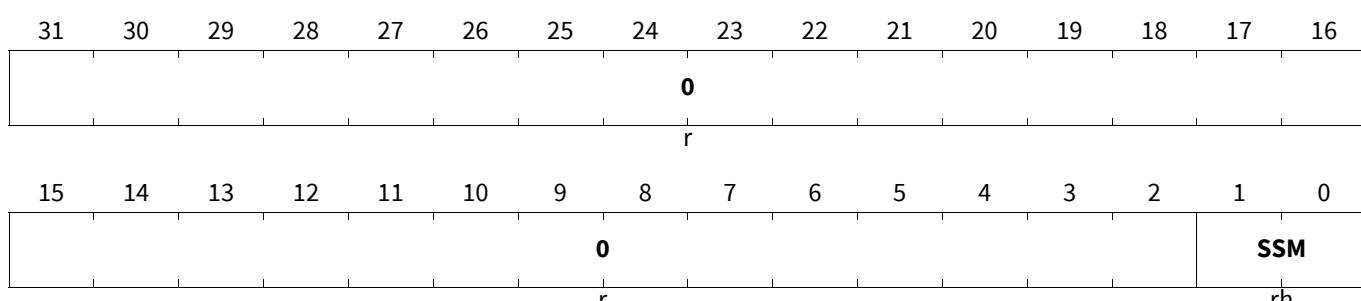
This register enables to observe some internal states of the SMU\_core hardware. Definition will be completed based on design information.

#### DBG

##### Debug Register

(038<sub>H</sub>)

PowerOn Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
SSM	1:0	rh	<b>Running state of the SMU_core State Machine</b> 00 <sub>B</sub> START state 01 <sub>B</sub> RUN state 10 <sub>B</sub> FAULT state 11 <sub>B</sub> unspecified state

## Safety Management Unit (SMU)

Field	Bits	Type	Description
0	31:2	r	<b>Reserved</b> Read as 0; should be written with 0

### Port Control

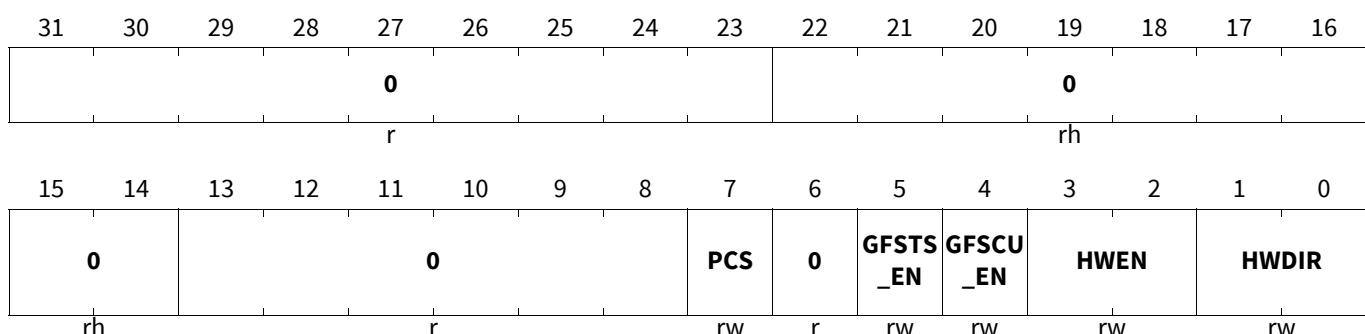
This register controls the connectivity with the Ports

#### PCTL

##### Port Control

(03C<sub>H</sub>)

PowerOn Reset Value: 00XX 8000<sub>H</sub>



Field	Bits	Type	Description
<b>HWDIR</b>	1:0	rw	<b>Port Direction.</b> This bitfield directly controls the value of the FSP_DIR output signal. Also refer to the General Purpose I/O Ports chapter for the HW_DIR signal specification.  $00_B$ sets FSP[0] and FSP[1] to input state $01_B$ sets FSP[0] to output state and FSP[1] to input state $10_B$ invalid input $11_B$ sets FSP[0] and FSP[1] to output state
<b>HWEN</b>	3:2	rw	<b>Port Enable</b> This bitfield directly controls the value of the FSP_EN output signal. When set to 11b the port output is directly driven by SMU_core (FSP[1:0]). Also refer to the General Purpose I/O Ports chapter for the HW_EN signal specification.  $00_B$ FSP[1:0] port output is not driven by SMU_core. $01_B$ FSP[0] port output is directly driven by SMU_core and FSP[1] port output is not driven by SMU_core. $10_B$ invalid input $11_B$ FSP[1:0] port output is directly driven by SMU_core
<b>GFSCU_EN</b>	4	rw	<b>Glitch Filter for ErrorPin SMU_FSP0 to SCU enable</b> $0_B$ Glitch Filter disabled $1_B$ Glitch Filter enabled
<b>GFSTS_EN</b>	5	rw	<b>Glitch Filter for ErrorPin SMU_FSP0 to register SMU_STS enable</b> $0_B$ Glitch Filter disabled $1_B$ Glitch Filter enabled

## Safety Management Unit (SMU)

Field	Bits	Type	Description
PCS	7	rw	<p><b>Pad Configuration Select</b>            This bit controls the latching of the SMU_core FSP (Error Pin) PAD configuration signals to ensure that upon an application reset or system reset the SMU_core FSP (Error Pin) PAD configuration is not affected. This field is only reset by power-on reset.</p> <ul style="list-style-type: none"> <li>Only with the first transition from 0 to 1 of this field the SMU_core FSP is operational. Any further configuration change in this bit field has no effect to the hardware.</li> <li>The fields HWDIR, HWEN and PCS shall be configured with a single software write command. Configuring each bit-field separately may lead to configuration inconsistencies. Refer to “<a href="#">Interface to the Ports (ErrorPin)</a>” on Page 7 for the overview of the SMU_core FSP (Error Pin) connectivity.</li> <li>The Error Pin Pad shall be configured to the targeted function in the Port control logic before the SMU_core takes over the control.</li> </ul> <p> <math>0_B</math> The PAD configuration is controlled by the PORT registers.  <math>1_B</math> The PAD configuration is controlled by the SMU.         </p>
0	6, 13:8, 31:23	r	<p><b>Reserved</b>            Read as 0; should be written with 0         </p>
0	22:14	rh	<p><b>Reserved</b>            Read as 0; should be written with 0         </p>

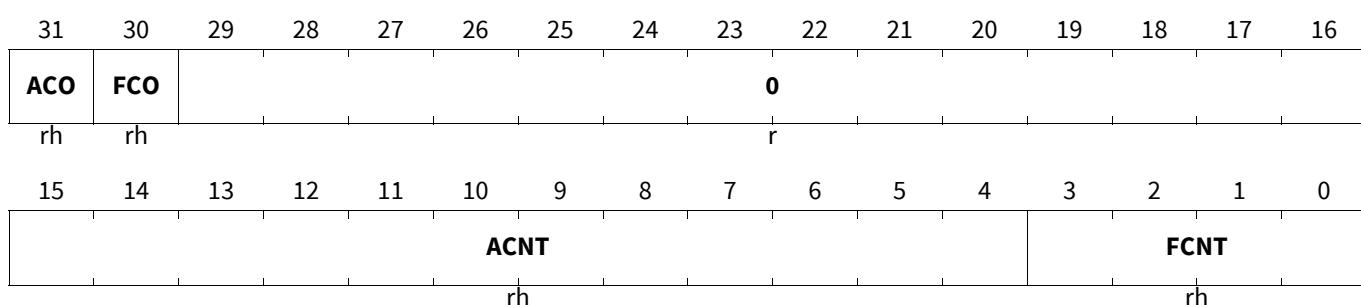
## Alarm and Fault Counter

### AFCNT

#### Alarm and Fault Counter

(040<sub>H</sub>)

PowerOn Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
FCNT	3:0	rh	<p><b>Fault Counter.</b>            This field is incremented by hardware when the SMU_core state machine goes from the RUN state to the FAULT state (see <a href="#">Figure 15.3.1.7 “SMU_core State Machine” on Page 22</a>). The counter value holds if the maximum value is reached.         </p>

## Safety Management Unit (SMU)

Field	Bits	Type	Description
<b>ACNT</b>	15:4	rh	<b>Alarm Counter.</b> This field is incremented by hardware when the SMU_core processes an internal action related to an alarm event (see <a href="#">Figure 15.3.1.5.3 “Alarm operation” on Page 15</a> ). The counter value holds if the maximum value is reached.
<b>FCO</b>	30	rh	<b>Fault Counter Overflow.</b> This bit is set by hardware if the FCNT counter reached the maximum value and an increment condition is present.
<b>ACO</b>	31	rh	<b>Alarm Counter Overflow.</b> This bit is set by hardware if the ACNT counter reached the maximum value and an increment condition is present.
<b>0</b>	29:16	r	<b>Reserved</b> Read as 0; should be written with 0.

### Recovery Timer 0 Alarm Configuration 0

Note: *It is possible to configure multiple times the same group identifier in GID0/1 fields.*

#### RTAC00

<b>Recovery Timer 0 Alarm Configuration 0 (060<sub>H</sub>) Application Reset Value: 00A8 0108<sub>H</sub></b>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								ALID1				GID1			
r								rw				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ALID0				GID0			
r								rw				rw			

Field	Bits	Type	Description
<b>GID0</b>	3:0	rw	<b>Group Index 0.</b> This field enables to specify if an alarm from this alarm group can use the recovery timer 0. The functionality of this field is described in <a href="#">“Recovery Timer” on Page 18</a> .
<b>ALID0</b>	8:4	rw	<b>Alarm Identifier 0.</b> This field specifies the alarm index related to the group index specified in GID0.
<b>GID1</b>	19:16	rw	<b>Group Index 1.</b> This field enables to specify if an alarm from this alarm group can use the recovery timer 0. The functionality of this field is described in <a href="#">“Recovery Timer” on Page 18</a> .
<b>ALID1</b>	24:20	rw	<b>Alarm Identifier 1.</b> This field specifies the alarm index related to the group index specified in GID1.

## Safety Management Unit (SMU)

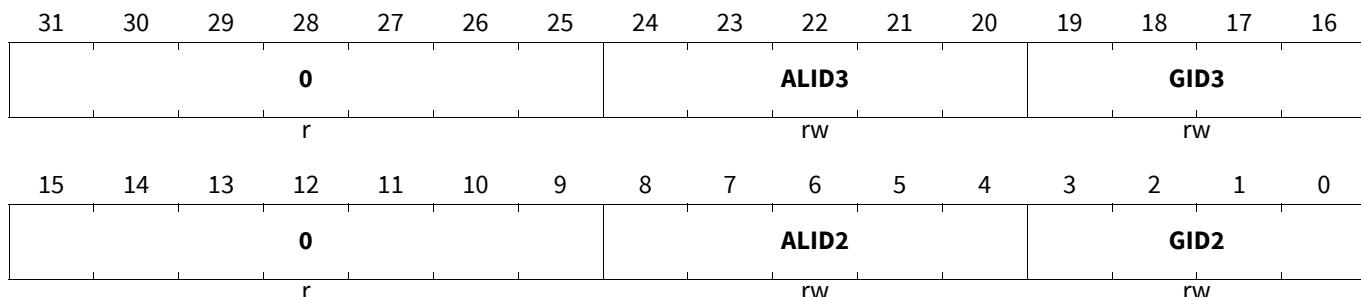
Field	Bits	Type	Description
0	15:9, 31:25	r	<b>Reserved</b> Read as 0; should be written with 0.

### Recovery Timer 0 Alarm Configuration 1

Note: *It is possible to configure multiple times the same group identifier in GID2/3 fields.*

#### RTAC01

**Recovery Timer 0 Alarm Configuration 1 (064<sub>H</sub>) Application Reset Value: 00C8 00B8<sub>H</sub>**



Field	Bits	Type	Description
<b>GID2</b>	3:0	rw	<b>Group Index 2.</b> This field enables to specify if an alarm from this alarm group can use the recovery timer 0. The functionality of this field is described in “ <a href="#">Recovery Timer” on Page 18.</a>
<b>ALID2</b>	8:4	rw	<b>Alarm Identifier 0.</b> This field specifies the alarm index related to the group index specified in GID2.
<b>GID3</b>	19:16	rw	<b>Group Index 3.</b> This field enables to specify if an alarm from this alarm group can use the recovery timer 3. The functionality of this field is described in “ <a href="#">Recovery Timer” on Page 18.</a>
<b>ALID3</b>	24:20	rw	<b>Alarm Identifier 1.</b> This field specifies the alarm index related to the group index specified in GID3.
0	15:9, 31:25	r	<b>Reserved</b> Read as 0; should be written with 0.

### Recovery Timer 1 Alarm Configuration 0

Note: *It is possible to configure multiple times the same group identifier in GID0/1 fields.*

**Safety Management Unit (SMU)****RTAC10****Recovery Timer 1 Alarm Configuration 0**(068<sub>H</sub>)**Application Reset Value: 00E8 00D8<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								ALID1				GID1			
r								rw				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ALID0				GID0			
r								rw				rw			

Field	Bits	Type	Description
<b>GID0</b>	3:0	rw	<b>Group Index 0.</b> This field enables to specify if an alarm from this alarm group can use the recovery timer 1. The functionality of this field is described in “ <a href="#">Recovery Timer” on Page 18.</a>
<b>ALID0</b>	8:4	rw	<b>Alarm Identifier 0.</b> This field specifies the alarm index related to the group index specified in GID0.
<b>GID1</b>	19:16	rw	<b>Group Index 1.</b> This field enables to specify if an alarm from this alarm group can use the recovery timer 1. The functionality of this field is described in “ <a href="#">Recovery Timer” on Page 18.</a>
<b>ALID1</b>	24:20	rw	<b>Alarm Identifier 1.</b> This field specifies the alarm index related to the group index specified in GID1.
<b>0</b>	15:9, 31:25	r	<b>Reserved</b> Read as 0; should be written with 0.

**Recovery Timer 1 Alarm Configuration 1**

*Note:* It is possible to configure multiple times the same group identifier in GID2/3 fields.

**RTAC11****Recovery Timer 1 Alarm Configuration 1**(06C<sub>H</sub>)**Application Reset Value: 00F8 00F8<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								ALID3				GID3			
r								rw				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ALID2				GID2			
r								rw				rw			

## Safety Management Unit (SMU)

Field	Bits	Type	Description
<b>GID2</b>	3:0	rw	<b>Group Index 2.</b> This field enables to specify if an alarm from this alarm group can use the recovery timer 1. The functionality of this field is described in “ <a href="#">Recovery Timer” on Page 18.</a>
<b>ALID2</b>	8:4	rw	<b>Alarm Identifier 2.</b> This field specifies the alarm index related to the group index specified in GID2.
<b>GID3</b>	19:16	rw	<b>Group Index 3.</b> This field enables to specify if an alarm from this alarm group can use the recovery timer 1. The functionality of this field is described in “ <a href="#">Recovery Timer” on Page 18.</a>
<b>ALID3</b>	24:20	rw	<b>Alarm Identifier 3.</b> This field specifies the alarm index related to the group index specified in GID3.
<b>0</b>	15:9, 31:25	r	<b>Reserved</b> Read as 0; should be written with 0.

### Alarm Executed Status Register

The Alarm Executed Status Register is used to show, which alarm mechanisms are executed.

#### AEX

#### Alarm Executed Status Register (070<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
					<b>EMSA EM</b>	<b>0</b>	<b>NMIAE M</b>	<b>RST5A EM</b>	<b>RST4A EM</b>	<b>RST3A EM</b>	<b>RST2A EM</b>	<b>RST1A EM</b>	<b>RST0A EM</b>	<b>IRQ2A EM</b>	<b>IRQ1A EM</b>	<b>IRQ0A EM</b>
r				rh	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
					<b>EMSSTS</b>	<b>0</b>	<b>NMIST S</b>	<b>RST5S TS</b>	<b>RST4S TS</b>	<b>RST3S TS</b>	<b>RST2S TS</b>	<b>RST1S TS</b>	<b>RST0S TS</b>	<b>IRQ2S TS</b>	<b>IRQ1S TS</b>	<b>IRQ0S TS</b>
r				rh	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	

Field	Bits	Type	Description
<b>IRQ0STS</b>	0	rh	<b>IRQ0 Request Status</b> This bit indicates whether a IRQ0 request was serviced or not. This bit is set by the SMU_core after a alarm configured for IRQ0 is detected. This bit can be cleared by writing with ‘1’ to the CLR bit in the related AEXCLR register. 0 <sub>B</sub> NO IRQ0 request was serviced 1 <sub>B</sub> IRQ0 request was serviced

## Safety Management Unit (SMU)

Field	Bits	Type	Description
<b>IRQ1STS</b>	1	rh	<p><b>IRQ1 Request Status</b></p> <p>This bit indicates whether a IRQ1 request was serviced or not. This bit is set by the SMU_core after a alarm configured for IRQ1 is detected.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> NO IRQ1 request was serviced 1<sub>B</sub> IRQ1 request was serviced</p>
<b>IRQ2STS</b>	2	rh	<p><b>IRQ2 Request Status</b></p> <p>This bit indicates whether a IRQ2 request was serviced or not. This bit is set by the SMU_core after a alarm configured for IRQ2 is detected.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> NO IRQ2 request was serviced 1<sub>B</sub> IRQ2 request was serviced</p>
<b>RST0STS</b>	3	rh	<p><b>RST0 Request Status</b></p> <p>This bit indicates whether a RST0 request was serviced or not. This bit is set by the SMU_core after a alarm configured for RST0 is detected.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> NO RST0 request was serviced 1<sub>B</sub> RST0 request was serviced</p>
<b>RST1STS</b>	4	rh	<p><b>RST1 Request Status</b></p> <p>This bit indicates whether a RST1 request was serviced or not. This bit is set by the SMU_core after a alarm configured for RST1 is detected.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> NO RST1 request was serviced 1<sub>B</sub> RST1 request was serviced</p>
<b>RST2STS</b>	5	rh	<p><b>RST2 Request Status</b></p> <p>This bit indicates whether a RST2 request was serviced or not. This bit is set by the SMU_core after a alarm configured for RST2 is detected.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> NO RST2 request was serviced 1<sub>B</sub> RST2 request was serviced</p>
<b>RST3STS</b>	6	rh	<p><b>RST3 Request Status</b></p> <p>This bit indicates whether a RST3 request was serviced or not. This bit is set by the SMU_core after a alarm configured for RST3 is detected.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> NO RST3 request was serviced 1<sub>B</sub> RST3 request was serviced</p>

**Safety Management Unit (SMU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>RST4STS</b>	7	rh	<p><b>RST4 Request Status</b></p> <p>This bit indicates whether a RST4 request was serviced or not. This bit is set by the SMU_core after a alarm configured for RST4 is detected.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> NO RST4 request was serviced 1<sub>B</sub> RST4 request was serviced</p>
<b>RST5STS</b>	8	rh	<p><b>RST5 Request Status</b></p> <p>This bit indicates whether a RST5 request was serviced or not. This bit is set by the SMU_core after a alarm configured for RST5 is detected.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> NO RST5 request was serviced 1<sub>B</sub> RST5 request was serviced</p>
<b>NMISTS</b>	9	rh	<p><b>NMI Request Status</b></p> <p>This bit indicates whether a NMI request was serviced or not. This bit is set by the SMU_core after a alarm configured for NMI is detected.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> NO NMI request was serviced 1<sub>B</sub> NMI request was serviced</p>
<b>EMSSTS</b>	11	rh	<p><b>EMS Request Status</b></p> <p>This bit indicates whether a EMS request, triggered by an alarm (not SMU_ActivatePES), was serviced or not. This bit is set by the SMU_core after a alarm configured for EMS is detected.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> NO EMS request was serviced 1<sub>B</sub> EMS request was serviced</p>
<b>IRQ0AEM</b>	16	rh	<p><b>IRQ0 AEM</b></p> <p>This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for IRQ0 Request where this alarm handler was blocked because of AEX.IRQ0STS.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> No alarm is missed 1<sub>B</sub> At least one alarm is missed</p>
<b>IRQ1AEM</b>	17	rh	<p><b>IRQ1 AEM</b></p> <p>This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for IRQ1 Request where this alarm handler was blocked because of AEX.IRQ1STS.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> No alarm is missed 1<sub>B</sub> At least one alarm is missed</p>

**Safety Management Unit (SMU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>IRQ2AEM</b>	18	rh	<p><b>IRQ2 AEM</b></p> <p>This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for IRQ2 Request where this alarm handler was blocked because of AEX.IRQ2STS.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> No alarm is missed 1<sub>B</sub> At least one alarm is missed</p>
<b>RST0AEM</b>	19	rh	<p><b>RST0 AEM</b></p> <p>This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for CPU RST0 Request where this alarm handler was blocked because of AEX.RST0STS.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> No alarm is missed 1<sub>B</sub> At least one alarm is missed</p>
<b>RST1AEM</b>	20	rh	<p><b>RST1 AEM</b></p> <p>This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for CPU RST1 Request where this alarm handler was blocked because of AEX.RST1STS.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> No alarm is missed 1<sub>B</sub> At least one alarm is missed</p>
<b>RST2AEM</b>	21	rh	<p><b>RST2 AEM</b></p> <p>This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for CPU RST2 Request where this alarm handler was blocked because of AEX.RST2STS.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> No alarm is missed 1<sub>B</sub> At least one alarm is missed</p>
<b>RST3AEM</b>	22	rh	<p><b>RST3 AEM</b></p> <p>This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for CPU RST3 Request where this alarm handler was blocked because of AEX.RST3STS.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> No alarm is missed 1<sub>B</sub> At least one alarm is missed</p>

## Safety Management Unit (SMU)

Field	Bits	Type	Description
<b>RST4AEM</b>	23	rh	<p><b>RST4 AEM</b></p> <p>This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for CPU RST4 Request where this alarm handler was blocked because of AEX.RST4STS.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> No alarm is missed 1<sub>B</sub> At least one alarm is missed</p>
<b>RST5AEM</b>	24	rh	<p><b>RST5 AEM</b></p> <p>This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for CPU RST5 Request where this alarm handler was blocked because of AEX.RST5STS.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> No alarm is missed 1<sub>B</sub> At least one alarm is missed</p>
<b>NMIAEM</b>	25	rh	<p><b>NMI AEM</b></p> <p>This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for NMI Request where this alarm handler was blocked because of AEX.NMISTS.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> No alarm is missed 1<sub>B</sub> At least one alarm is missed</p>
<b>EMSAEM</b>	27	rh	<p><b>EMS AEM</b></p> <p>This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for EMS Request where this alarm handler was blocked because of AEX.EMSSTS.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related AEXCLR register.</p> <p>0<sub>B</sub> No alarm is missed 1<sub>B</sub> At least one alarm is missed</p>
<b>0</b>	10, 15:12, 26, 31:28	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

### Alarm Executed Status Clear Register

The Alarm Executed Status Clear register is used to clear the Alarm Executed Status(SMU\_AEX.xx).

**Safety Management Unit (SMU)****AEXCLR****Alarm Executed Status Clear Register**(074<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
					EMSA EMCL R	0	NMIAE MCLR	RST5A EMCL R	RST4A EMCL R	RST3A EMCL R	RST2A EMCL R	RST1A EMCL R	RST0A EMCL R	IRQ2A EMCL R	IRQ1A EMCL R	IRQ0A EMCL R
r				w	r	w	w	w	w	w	w	w	w	w	w	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				EMSCL R	0	NMICL R	RST5C LR	RST4C LR	RST3C LR	RST2C LR	RST1C LR	RST0C LR	IRQ2C LR	IRQ1C LR	IRQ0C LR
r				w	r	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
<b>IRQ0CLR</b>	0	w	<b>IRQ0 Request Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear IRQ0 request status AEX.IRQ0STAT
<b>IRQ1CLR</b>	1	w	<b>IRQ1 Request Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear IRQ1 request status AEX.IRQ1STAT
<b>IRQ2CLR</b>	2	w	<b>IRQ2 Request Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear IRQ2 request status AEX.IRQ2STAT
<b>RST0CLR</b>	3	w	<b>RST0 Request Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear RST0 request status AEX.RST0STAT
<b>RST1CLR</b>	4	w	<b>RST1 Request Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear RST1 request status AEX.RST1STAT
<b>RST2CLR</b>	5	w	<b>RST2 Request Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear RST2 request status AEX.RST2STAT
<b>RST3CLR</b>	6	w	<b>RST3 Request Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear RST3 request status AEX.RST3STAT

## Safety Management Unit (SMU)

Field	Bits	Type	Description
<b>RST4CLR</b>	7	w	<b>RST4 Request Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear RST4 request status AEX.RST4STAT
<b>RST5CLR</b>	8	w	<b>RST5 Request Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear RST5 request status AEX.RST5STAT
<b>NMICLR</b>	9	w	<b>NMI Request Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear NMI request status AEX.NMISTAT
<b>EMSCLR</b>	11	w	<b>EMS Request Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear EMS request status AEX.EMSSTAT
<b>IRQ0AEMCLR</b>	16	w	<b>IRQ0 AEM Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear AEM status AEX.IRQ0AEM
<b>IRQ1AEMCLR</b>	17	w	<b>IRQ1 AEM Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear AEM status AEX.IRQ1AEM
<b>IRQ2AEMCLR</b>	18	w	<b>IRQ2 AEM Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear AEM status AEX.IRQ2AEM
<b>RST0AEMCLR</b>	19	w	<b>RST0 AEM Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear AEM status AEX.RST0AEM
<b>RST1AEMCLR</b>	20	w	<b>RST1 AEM Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear AEM status AEX.RST1AEM
<b>RST2AEMCLR</b>	21	w	<b>RST2 AEM Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear AEM status AEX.RST2AEM
<b>RST3AEMCLR</b>	22	w	<b>RST3 AEM Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear AEM status AEX.RST3AEM

**Safety Management Unit (SMU)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>RST4AEMCLR</b>	23	w	<b>RST4 AEM Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear AEM status AEX.RST4AEM
<b>RST5AEMCLR</b>	24	w	<b>RST5 AEM Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear AEM status AEX.RST5AEM
<b>NMIAEMCLR</b>	25	w	<b>NMI AEM Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear AEM status AEX.NMIAEM
<b>EMSAEMCLR</b>	27	w	<b>EMS AEM Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear AEM status AEX.EMSAEM
<b>0</b>	10, 15:12, 26, 31:28	r	<b>Reserved</b> Read as 0; should be written with 0.

## Safety Management Unit (SMU)

## 15.4.1.3 SMU\_core Alarm Configuration Registers

## Alarm Configuration Register

AGiCFj (i=0-11;j=0-2)

## Alarm Configuration Register

(100<sub>H</sub>+i\*12+j\*4)Reset Value: [Table 541](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw															

Field	Bits	Type	Description
CFz (z=0-31)	z	rw	<b>Configuration flag x (x=0-2) for alarm z belonging to alarm group i.</b> The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU_core when a fault state is reported by the alarm n belonging to this group. 0 <sub>B</sub> Configuration flag x (x=0-2) is set to 0 1 <sub>B</sub> Configuration flag x (x=0-2) is set to 1

**Table 541 Reset Values of AGiCFj (i=0-11;j=0-2)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Reset value for all alarm configuration registers except AG8CFG0, AG8CFG2, AG10CFG1 and AG10CFG2
Application Reset	0001 FC00 <sub>H</sub>	Reset value for AG8CFG0 and AG8CFG2
Application Reset	0003 0000 <sub>H</sub>	Reset value for AG10CFG1 and AG10CFG2

## 15.4.1.4 SMU\_core Alarm Configuration Registers (Fault Signaling Protocol)

## SMU\_core FSP Configuration Register

AGiFSP (i=0-11)

## SMU\_core FSP Configuration Register

(190<sub>H</sub>+i\*4)Reset Value: [Table 542](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FE31	FE30	FE29	FE28	FE27	FE26	FE25	FE24	FE23	FE22	FE21	FE20	FE19	FE18	FE17	FE16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE15	FE14	FE13	FE12	FE11	FE10	FE9	FE8	FE7	FE6	FE5	FE4	FE3	FE2	FE1	FE0
rw															

**Safety Management Unit (SMU)**

Field	Bits	Type	Description
<b>FEz (z=0-31)</b>	z	rw	<p><b>Fault signaling configuration flag for alarm z belonging to alarm group i.</b></p> <p>0<sub>B</sub> FSP disabled for this alarm event 1<sub>B</sub> FSP enabled for this alarm event</p>

**Table 542 Reset Values of AGiFSP (i=0-11)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Reset value for all alarm FSP configuration registers except AG10FSP
Application Reset	0003 0000 <sub>H</sub>	Reset value for AG10FSP

## Safety Management Unit (SMU)

### 15.4.1.5 SMU\_core Alarm Status Registers

#### Alarm Status Register

Refer to Alarm Status Registers for the conditions to set and reset the status flag by software.

#### AGi (i=0-11)

Alarm Status Register $(1C0_H+i*4)$																Application Reset Value: 0000 0000 <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>SF31</b>	<b>SF30</b>	<b>SF29</b>	<b>SF28</b>	<b>SF27</b>	<b>SF26</b>	<b>SF25</b>	<b>SF24</b>	<b>SF23</b>	<b>SF22</b>	<b>SF21</b>	<b>SF20</b>	<b>SF19</b>	<b>SF18</b>	<b>SF17</b>	<b>SF16</b>	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	
<b>SF15</b>	<b>SF14</b>	<b>SF13</b>	<b>SF12</b>	<b>SF11</b>	<b>SF10</b>	<b>SF9</b>	<b>SF8</b>	<b>SF7</b>	<b>SF6</b>	<b>SF5</b>	<b>SF4</b>	<b>SF3</b>	<b>SF2</b>	<b>SF1</b>	<b>SF0</b>	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	

Field	Bits	Type	Description
<b>SFz (z=0-31)</b>	z	rwh	<b>Status flag for alarm z belonging to alarm group i.</b> 0 <sub>B</sub> Status flag z does not report a fault condition 1 <sub>B</sub> Status flag z reports a fault condition

### 15.4.1.6 SMU\_core Alarm Diagnosis Registers

#### Alarm Debug Register

Note: Writing to this register has no effect

#### ADI (i=0-11)

Alarm Debug Register $(200_H+i*4)$																PowerOn Reset Value: 0000 0000 <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>DF31</b>	<b>DF30</b>	<b>DF29</b>	<b>DF28</b>	<b>DF27</b>	<b>DF26</b>	<b>DF25</b>	<b>DF24</b>	<b>DF23</b>	<b>DF22</b>	<b>DF21</b>	<b>DF20</b>	<b>DF19</b>	<b>DF18</b>	<b>DF17</b>	<b>DF16</b>	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	
<b>DF15</b>	<b>DF14</b>	<b>DF13</b>	<b>DF12</b>	<b>DF11</b>	<b>DF10</b>	<b>DF9</b>	<b>DF8</b>	<b>DF7</b>	<b>DF6</b>	<b>DF5</b>	<b>DF4</b>	<b>DF3</b>	<b>DF2</b>	<b>DF1</b>	<b>DF0</b>	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	

Field	Bits	Type	Description
<b>DFz (z=0-31)</b>	z	rh	<b>Diagnosis flag for alarm z belonging to alarm group i.</b> The diagnosis registers make a snapshot of the alarm group status registers when either the executed alarm action is a reset or a state machine transition to FAULT state takes place. 0 <sub>B</sub> Status flag z does not report a fault condition 1 <sub>B</sub> Status flag z reports a fault condition

**Safety Management Unit (SMU)****15.4.1.7 SMU\_core Special Safety Registers: Register Monitor****Register Monitor Control****RMCTL**

Register Monitor Control (300 <sub>H</sub> )																Application Reset Value: 0000 0000 <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	0	0	0	0	TE10	TE9	TE8	TE7	TE6	TE5	TE4	TE3	TE2	TE1	TE0	
																r	r	r	r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

Field	Bits	Type	Description
TEz (z=0-10)	z	rw	<p><b>Test Enable.</b>  <b>This bit controls the test of the register monitor safety mechanism. Setting this bit starts the self-test of the safety flip-flops in the corresponding module.</b></p> <p>0<sub>B</sub> 0 Test mode disabled  1<sub>B</sub> 1 Test mode enabled</p>
0	31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, 16, 15, 14, 13, 12, 11	r	<p><b>Reserved</b>  Read as 0; should be written with 0.</p>

**Register Monitor Error Flags****RMEF**

Register Monitor Error Flags (304 <sub>H</sub> )																Application Reset Value: 0000 0000 <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	0	0	0	0	EF10	EF9	EF8	EF7	EF6	EF5	EF4	EF3	EF2	EF1	EF0	
																r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	

**Safety Management Unit (SMU)**

Field	Bits	Type	Description
<b>EFz (z=0-10)</b>	z	rwh	<p><b>Status flag related to the different instances of the register monitor safety mechanism.</b></p> <p><b>It reports a real flip flop failure in non-test mode as well as an unexpected behavior in test-mode.</b></p> <p>This flag can only be cleared by software, a set by software has no effect</p> <p><math>0_B</math> Error flag z does not report a fault condition</p> <p><math>1_B</math> Error flag z reports a fault condition</p>
<b>0</b>	31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, 16, 15, 14, 13, 12, 11	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Register Monitor Self Test Status****RMSTS**
**Register Monitor Self Test Status** **(308<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>STS10</b>	<b>STS9</b>	<b>STS8</b>	<b>STS7</b>	<b>STS6</b>	<b>STS5</b>	<b>STS4</b>	<b>STS3</b>	<b>STS2</b>	<b>STS1</b>	<b>STS0</b>
r	r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
<b>STSz (z=0-10)</b>	z	rwh	<p><b>Ready flag related to the different instances of the register monitor safety mechanism.</b></p> <p>A logical ‘1’ of this bit indicates that the register monitor test has been executed. This bit can only be cleared by software, a set by software has no effect.</p> <p><math>0_B</math> Self-test has not completed</p> <p><math>1_B</math> Self-test has completed</p>
<b>0</b>	31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, 16, 15, 14, 13, 12, 11	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

## Safety Management Unit (SMU)

### 15.4.2 SMU\_stdby Module Registers

**Table 543** lists all registers implemented in the SMU\_stdby.

For information about the address space of the SMU\_stdby registers please refer to the Power Management System chapter.

**Table 543 Register Overview - SMU\_STDBY (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
AG2i_STDBY	Alarm Status Register	188 <sub>H</sub> +i*4	U,SV	SV,SE,P	LVD Reset	<b>73</b>
MONBISTSTAT	SMU_stdby BIST Status Register	190 <sub>H</sub>	U,SV	BE	See page <b>75</b>	<b>75</b>
MONBISTCTRL	SMU_stdby BIST Control Register	198 <sub>H</sub>	U,SV	SV,SE,P	See page <b>74</b>	<b>74</b>
CMD_STDBY	SMU_stdby Command Register	19C <sub>H</sub>	U,SV	SV,SE,P	See page <b>69</b>	<b>69</b>
AG2iFSP_STDBY	SMU_stdby FSP Configuration Register	1A4 <sub>H</sub> +i*4	U,SV	SV,SE,P	See page <b>71</b>	<b>71</b>

#### 15.4.2.1 SMU\_stdby Command Register

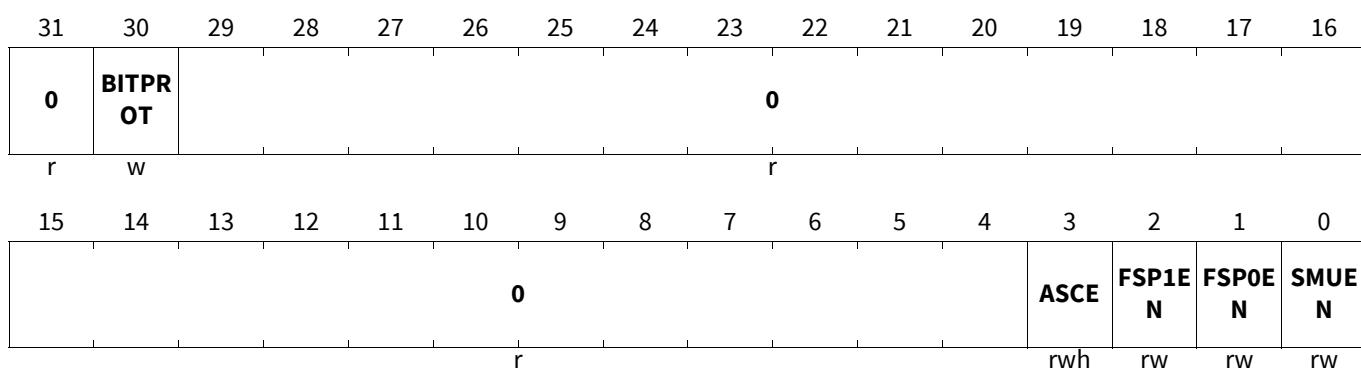
##### SMU\_stdby Command Register

###### CMD\_STDBY

###### SMU\_stdby Command Register

(19C<sub>H</sub>)

Reset Value: **Table 545**



<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>SMUEN</b>	0	rw	<b>SMU_stdby Module Enable</b> This bit enables SMU_stdby to issue a FSP reaction when an alarm is received. Also, SMUEN needs to be set to enter the SMU_stdby BIST mode. 0 <sub>B</sub> SMU_stdby disabled. 1 <sub>B</sub> SMU_stdby enabled.

## Safety Management Unit (SMU)

Field	Bits	Type	Description
<b>FSP0EN</b>	1	rw	<b>SMU_stby FSP0 Error pin enable</b> This bit enables SMU_stby Error pin function to be able set P33.8 to fault state. $0_B$ SMU_stby Error Pin fault indication function on P33.8 inactive. $1_B$ SMU_stby Error Pin fault indication function on P33.8 active.
<b>FSP1EN</b>	2	rw	<b>SMU_stby FSP1 Error pin enable</b> This bit enables SMU_stby Error pin function to be able set P33.10 to fault state. $0_B$ SMU_stby Error Pin fault indication function on P33.10 inactive. $1_B$ SMU_stby Error Pin fault indication function on P33.10 active.
<b>ASCE</b>	3	rwh	<b>SMU_stby alarm status clear enable</b> This bit controls if a status flag set in an AGx register upon detection of the alarm event can be cleared by software or not. When ASCE is enabled, software shall write a 1 to bit position in AGx to clear the bit (W1C). When a W1C action takes place the ASCE bit is automatically cleared to 0 by hardware and software shall set the ASCE bit again. $0_B$ SMU_stby alarm status bits in AG2i_STDBY cannot be cleared. $1_B$ SMU_stby alarm status bits in AG2i_STDBY can be cleared
<b>BITPROT</b>	30	w	<b>CMD_STDBY register bits protection</b> Setting this bit enables that bits SMUEN, FSP0EN, FSP1EN or/and ASCE can be changed in this write operation. This bit is read as zero.
<b>0</b>	29:4, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 544 Access Mode Restrictions of CMD\_STDBY sorted by descending priority**

Mode Name	Access Mode		Description
otherwise	r	FSP0EN, FSP1EN, SMUEN	
	rh	ASCE	
write 1 to BITPROT (default)	rw	FSP0EN, FSP1EN, SMUEN	
	rwh	ASCE	

**Table 545 Reset Values of CMD\_STDBY**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Warm PORST	0000 0000 <sub>H</sub>	

**Safety Management Unit (SMU)****15.4.2.2 SMU\_stdby Alarm Configuration Register (Fault Signaling Protocol)****SMU\_stdby FSP Configuration Register****AG2iFSP\_STDBY (i=0)****SMU\_stdby FSP Configuration Register (1A4<sub>H</sub>+i\*4) Reset Value: Table 547**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	<b>BITPR OT</b>							0						0	
r	w							r						r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>FE15</b>	<b>FE14</b>	<b>FE13</b>	<b>FE12</b>	<b>FE11</b>	<b>FE10</b>	<b>FE9</b>	<b>FE8</b>	<b>FE7</b>	<b>FE6</b>	<b>FE5</b>	<b>FE4</b>	0	0	0	0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	r	r	r	r

Field	Bits	Type	Description
<b>FEz (z=4-15)</b>	z	rw	<b>Fault signaling configuration flag for alarm z belonging to alarm group i.</b> 0 <sub>B</sub> FSP disabled for this alarm event. 1 <sub>B</sub> FSP enabled for this alarm event.
<b>BITPROT</b>	30	w	<b>AG2iFSP_STDBY register bits protection</b> Setting this bit enables that bits FE(z) can be changed in this write operation. This bit is read as zero.
0	16, 3, 2, 1, 0, 29:17, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 546 Access Mode Restrictions of AG2iFSP\_STDBY (i=0) sorted by descending priority**

Mode Name	Access Mode		Description
otherwise	r	FEz (z=4-15)	
write 1 to <b>BITPROT</b> (default)	rw	FEz (z=4-15)	

**Table 547 Reset Values of AG2iFSP\_STDBY (i=0)**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Warm PORST	0000 0000 <sub>H</sub>	

**Safety Management Unit (SMU)****AG2iFSP\_STDBY (i=1)****SMU\_stby FSP Configuration Register**(1A4<sub>H</sub>+i\*4)Reset Value: [Table 549](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>0</b>	<b>BITPR OT</b>							<b>0</b>							<b>FE16</b>
r	w							r							rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>FE15</b>	<b>FE14</b>	<b>FE13</b>	<b>FE12</b>	<b>FE11</b>	<b>FE10</b>	<b>FE9</b>	<b>FE8</b>	<b>FE7</b>	<b>0</b>	<b>FE5</b>	<b>FE4</b>	<b>FE3</b>	<b>FE2</b>	<b>FE1</b>	<b>FE0</b>
rw	rw	rw	rw	rw	rw	rw	rw	rw	r	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
<b>FEz (z=0-5,7-16)</b>	z	rw	<b>Fault signaling configuration flag for alarm z belonging to alarm group i.</b> 0 <sub>B</sub> FSP disabled for this alarm event. 1 <sub>B</sub> FSP enabled for this alarm event.
<b>BITPROT</b>	30	w	<b>AG2iFSP_STDBY register bits protection</b> Setting this bit enables that bits FE(z) can be changed in this write operation. This bit is read as zero.
<b>0</b>	6, 29:17, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 548 Access Mode Restrictions of AG2iFSP\_STDBY (i=1) sorted by descending priority**

Mode Name	Access Mode		Description
otherwise	r	FEz (z=0-5,7-16)	
write 1 to <b>BITPROT</b> (default)	rw	FEz (z=0-5,7-16)	

**Table 549 Reset Values of AG2iFSP\_STDBY (i=1)**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Warm PORST	0000 0000 <sub>H</sub>	

**Safety Management Unit (SMU)****15.4.2.3 SMU\_stdby Alarm Status Register****Alarm Status Register****AG2i\_STDBY (i=0)****Alarm Status Register**(188<sub>H</sub>+i\*4)LVD Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	FSPER R							0						0	
r	rwh							r						r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	SF14	SF13	SF12	SF11	SF10	SF9	SF8	SF7	SF6	SF5	SF4	0	0	0	0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	r	r	r	r

Field	Bits	Type	Description
SFz (z=4-15)	z	rwh	<b>Status flag for alarm z belonging to alarm group i.</b> 0 <sub>B</sub> Status flag z does not report a fault condition 1 <sub>B</sub> Status flag z reports a fault condition
FSPERR	30	rwh	<b>Error Pin Fault State Status Bit</b> The bit indicates that Error pin was set into fault state by the SMU_stdby. Reset by setting to 1. If the Error Pins were set in fault state by the SMU_stdby, resetting this bit sets the Error Pins back in fault free state 0 <sub>B</sub> Error pin was not set into fault state 1 <sub>B</sub> The Error pin was set into fault state.
0	16, 3, 2, 1, 0, 29:17, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 550 Access Mode Restrictions of AG2i\_STDBY (i=0) sorted by descending priority**

Mode Name	Access Mode		Description
CMD_STDBY.ASCE = 1	rwh	FSPERR, SFz (z=4-15)	
(default)	rh	FSPERR, SFz (z=4-15)	

**AG2i\_STDBY (i=1)****Alarm Status Register**(188<sub>H</sub>+i\*4)LVD Reset Value: 0000 0000<sub>H</sub>

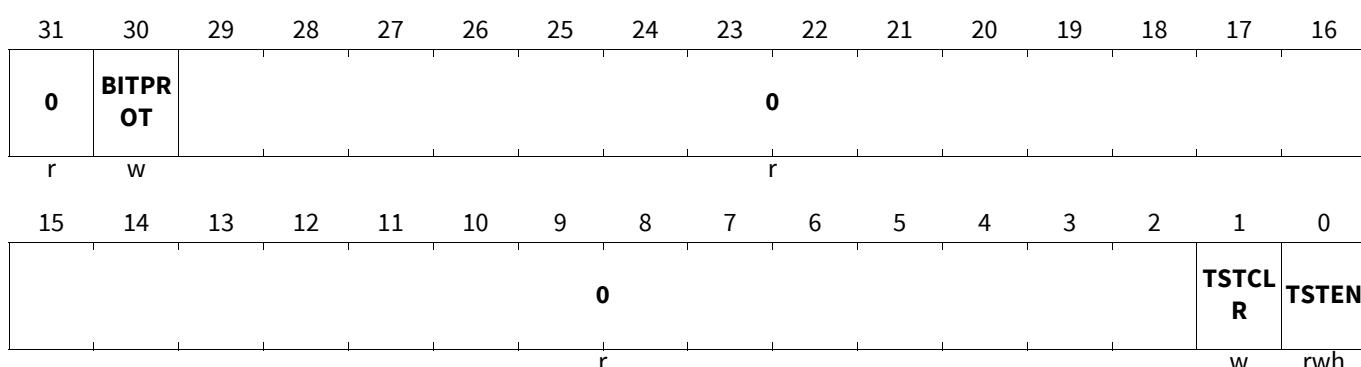
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0							0							
r	r							r						rwh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	SF14	SF13	SF12	SF11	SF10	SF9	SF8	SF7	0	SF5	SF4	SF3	SF2	SF1	SF0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	r	rwh	rwh	rwh	rwh	rwh	rwh

**Safety Management Unit (SMU)**

Field	Bits	Type	Description
<b>SFz (z=0-5,7-16)</b>	z	rwh	<b>Status flag for alarm z belonging to alarm group i.</b> 0 <sub>B</sub> Status flag z does not report a fault condition 1 <sub>B</sub> Status flag z reports a fault condition
<b>0</b>	6, 29:17, 30, 31	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 551 Access Mode Restrictions of AG2i\_STDBY (i=1) sorted by descending priority**

Mode Name	Access Mode		Description
CMD_STDBY.ASCE = 1	rwh	SFz (z=0-5,7-16)	
(default)	rh	SFz (z=0-5,7-16)	

**15.4.2.4 SMU\_stdby BIST Control Register****SMU\_stdby BIST Control Register****MONBISTCTRL****SMU\_stdby BIST Control Register (198<sub>H</sub>) Reset Value: Table 553**

Field	Bits	Type	Description
<b>TSTEN</b>	0	rwh	<b>SMU_stdby alarm BIST enable</b> If SMUEN in the CMD_stdby register is set to 1, setting the TSTEN bit triggers SMU_stdby BIST tests to test the alarm path and PMS components. This bit can only be changed if bit BITPROT is set in parallel. This bit is cleared by hardware at the end of the SMU_stdby BIST operation. 0 <sub>B</sub> SMU_stdby BIST test is inactive 1 <sub>B</sub> SMU_stdby BIST test triggered. When the SMU_stdby BIST is activated, the alarm signals tested by the bist are not propagated to the SMU_core.

**Safety Management Unit (SMU)**

Field	Bits	Type	Description
TSTCLR	1	w	<p><b>SMU_stdby BIST flag clear</b>  Setting this bit enables the clearing of the bitfields TSTOK, TSTDONE, TSTRUN, SMUERR and PMSERR of register MONBISTSTAT.</p> <p>0<sub>B</sub> No action  1<sub>B</sub> The bitfields TSTOK, TSTDONE, TSTRUN, SMUERR and PMSERR of register MONBISTSTAT are cleared. Before triggering SMU_stdby BIST mechanism via TSTEN, previous test results need to be cleared via TSTCLR.</p>
BITPROT	30	w	<p><b>Bit Protection TSTEN</b>  Setting this bit enables that bit TSTEN can be changed in this write operation. This bit is read as zero.</p>
0	29:2, 31	r	<p><b>Reserved</b>  Read as 0; should be written with 0.</p>

**Table 552 Access Mode Restrictions of MONBISTCTRL sorted by descending priority**

Mode Name	Access Mode		Description
otherwise	rh	TSTEN	
write 1 to BITPROT (default)	rwh	TSTEN	

**Table 553 Reset Values of MONBISTCTRL**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Warm PORST	0000 0000 <sub>H</sub>	

**15.4.2.5 SMU\_stdby BIST Status Register****SMU\_stdby BIST Status Register**

<b>MONBISTSTAT</b> <b>SMU_stdby BIST Status Register</b> <span style="float: right;">Reset Value: <a href="#">Table 554</a></span>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>0</b>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>0</b>										<b>PMSE RR</b>	<b>SMUE RR</b>	<b>TSTD ONE</b>	<b>TSTRU N</b>	<b>0</b>	<b>TSTOK</b>

## Safety Management Unit (SMU)

Field	Bits	Type	Description
TSTOK	0	rh	<b>SMU_stdby BIST ok bit</b> This status bit indicates that MONBIST test was successful and the result is as expected. 0 <sub>B</sub> The MONBIST test result is not ok. 1 <sub>B</sub> The MONBIST test result is ok.
TSTRUN	2	rh	<b>SMU_stdby BIST run bit</b> The status bit indicates that MONBIST test is ongoing. 0 <sub>B</sub> The MONBIST test is currently inactive. 1 <sub>B</sub> The MONBIST test is active.
TSTDONE	3	rh	<b>SMU_stdby BIST done bit</b> The status bit indicates that MONBIST test is completed. 0 <sub>B</sub> The MONBIST test is not started. 1 <sub>B</sub> The MONBIST test is done.
SMUERR	4	rh	<b>Error found in SMU_stdby found by SMU_stdby BIST</b> This status bit indicates that the MONBIST test found an error in the SMU_stdby. 0 <sub>B</sub> No error happened in SMU_STDBY module during MONBIST test. 1 <sub>B</sub> This bit is set if MONBIST test failed and the error occurred in SMU_STDBY module and alarm processing path. This status bit is cleared by setting TSTCLR bit..
PMSERR	5	rh	<b>Error found in PMS SARADC by SMU_stdby BIST</b> This status bit indicates that SMU_stdby BIST found an error in the PMS SARADC. 0 <sub>B</sub> No error happened in PMS module during MONBIST test. 1 <sub>B</sub> This bit is set if MONBIST test failed and the error occurred in Secondary voltage monitor and alarm generation in PMS. This status bit is cleared by setting TSTCLR bit.
0	1, 31:6	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 554 Reset Values of MONBISTSTAT**

Reset Type	Reset Value	Note
LVD Reset	0000 0000 <sub>H</sub>	
Warm PORST	0000 0000 <sub>H</sub>	

## 15.5 Revision History

**Table 555 Revision History**

Reference	Change to Previous Version	Comment
V4.0.17	No changes	
V4.0.18		
Page 73	Modified description of FSPERR bitfield	

**Safety Management Unit (SMU)****Table 555 Revision History (cont'd)**

<b>Reference</b>	<b>Change to Previous Version</b>	<b>Comment</b>
<a href="#">Page 24</a>	Removed FSP timing internal implementation details	
<a href="#">Page 18</a> and <a href="#">Page 24</a> and <a href="#">Page 46</a> and <a href="#">Page 49</a>	Added limitation with regard to usage of FSP and RTC register. Added guidelines for configuration of FSP	
<a href="#">Page 13</a>	Updated guideline for configuration of Error Pin	
<b>V4.0.19</b>		
<a href="#">Page 46</a>	Typo “Mhz” corrected to “MHz” in PRE1 field description of register FSP.	
<a href="#">Page 76</a>	Revision History of V4.0.18 is modified	
<b>V4.0.20</b>		
-	No functional changes.	
<b>V4.0.21</b>		
-	No functional changes.	

## Interrupt Router (IR)

# 16 Interrupt Router (IR)

The chapter describes the Interrupt Router module that schedules interrupts (here called service requests) from external resources, internal resources and SW to the CPU and the DMA modules (here called Service Provider).

### Scope of this Document

This document is valid for the TC3xx and covers the topics:

- Interrupt system architecture
- Interrupt system configuration
- Interrupt Router operations

### 16.1 Feature List

The following list shows the main features of the interrupt router module:

- Interrupt System with support of up to 1024 service requests
- Support of up to 255 service request priority levels per ICU<sup>1)</sup> / Service Provider
- Support of up to 8 ICUs / Service Providers
- A dedicated ICU for each implemented CPU / DMA module (Service Provider)
- Low latency arbitration - three / four clocks<sup>2)</sup> from receipt of an service request to sending it to the service provider
- Each peripheral interrupt with a dedicated Service Request Node (SRN)
- Each SRN with a programmable 8-bit priority vector<sup>1)</sup>
- Each SRN can be mapped to one of the implemented ICUs / Service Providers
- SRNs are cleared automatically by hardware on interrupt acknowledge by the configured service provider
- Interrupt System with Integrity support
- 8 General Purpose Service Requests (GPSR) per CPU that can be used as Software Interrupts (not assigned to peripherals or external interrupts)
- Service Request Broadcast Registers (SRB) to signal General Purpose Service Requests (Software Interrupts) simultaneously to multiple Service Providers
- Priority dependent masking of service requests (for CPUs, related control registers included in the CPUs)
- External Interrupts with filter modes and trigger modes (to e.g. falling edge, rising edge, high or low level). Modes can be configured during runtime<sup>3)</sup>
- CPU wake up support (service request to CPUx is signalled to SCU to wake up CPUx in case CPUx is in IDLE state)

### 16.2 Delta to TC2xx

The following functional changes where introduced from Aurix to TC3xx:

- General Purpose Service Request Group: Number of Service Request Nodes per group changed from 4 to 8 SRNs (see “[General Purpose Service Requests, Service Request Broadcast](#)” on Page 19)

1) Max. 255 of the implemented service requests can be mapped to one CPU

2) Depends on the complexity and the clock frequency of the Interrupt Router. Details for the implementation are described in the chapter Module Implementation

3) External Interrupt logic and related control registers are described in the System Control Unit (SCU) chapter, External Request Unit (ERU)

## Interrupt Router (IR)

- Broadcast Register: Broadcast register bit map was adapted to new GPSR SRN number (see “[General Purpose Service Requests, Service Request Broadcast” on Page 19](#))
- Broadcast Register: Each Broadcast register now with a dedicated ACCEN register (see “[Access protection of SRBx registers \(ACCEN\\_SRBy\)](#)” on Page 20)
- SRCx: Access protection of SRCx[31:0] was introduced (one ACCEN register per implemented TOS encoding/ICU, see “[Protection of the SRC Registers” on Page 7](#) )
- SRC.ECC: Size of ECC bit field was changed from 6 to 5 bit (see “[General Service Request Control Register Format](#)” on Page 4)
- SRC.TOS: Size of TOS bit field was changed to 3 bit (see “[General Service Request Control Register Format](#)” on Page 4)
- SRC offset and Index numbering scheme changed: SRCs are mapped to a 1024 \* 32 bit range. The Index Number (0-1023) is equal to the 32 bit offset of the SRC (see “[General Service Request Control Register Format](#)” on Page 4)
- On detection of a service request signaled to a Reserved TOS, an alarm is signaled to the SMU
- Mapping of SRC registers to the IR address map was re-worked

### 16.3 Overview

An interrupt request can be serviced either by the CPUs or by a DMA module. Interrupt requests are called “service requests” rather than “interrupt requests” in this document because they can be serviced by either one of the service providers.

The interrupt system is realized in the Interrupt Router module which includes the Service Request Nodes (SRNs), the Interrupt Control Units (ICUs) and additional functionality for SW development support.

As shown in [Figure 172](#), each module that can generate service requests is connected to one or more Service Request Nodes (SRNs) in the central Interrupt Router module. The Interrupt Router module includes also several general purpose Service Request Nodes (SRNs) that can be used for software (SW) triggered service requests.

Each SRN contains a Service Request Control Register (SRC) to configure the service request regarding e.g. priority, mapping to one of the available service providers.

Each SRN is connected to all ICUs in the interrupt router module where the SRNs control register setting defines the target service provider and the priority of the service request.

Each ICU handles the interrupt arbitration among competing service requests from SRNs that are mapped to the ICU.

Each ICU is connected to one service provider (CPU or DMA module) where the ICU offers the valid winning service request/SRN of an arbitration round and the service provider signals back to the ICU when and which service request it is processing.

## Interrupt Router (IR)

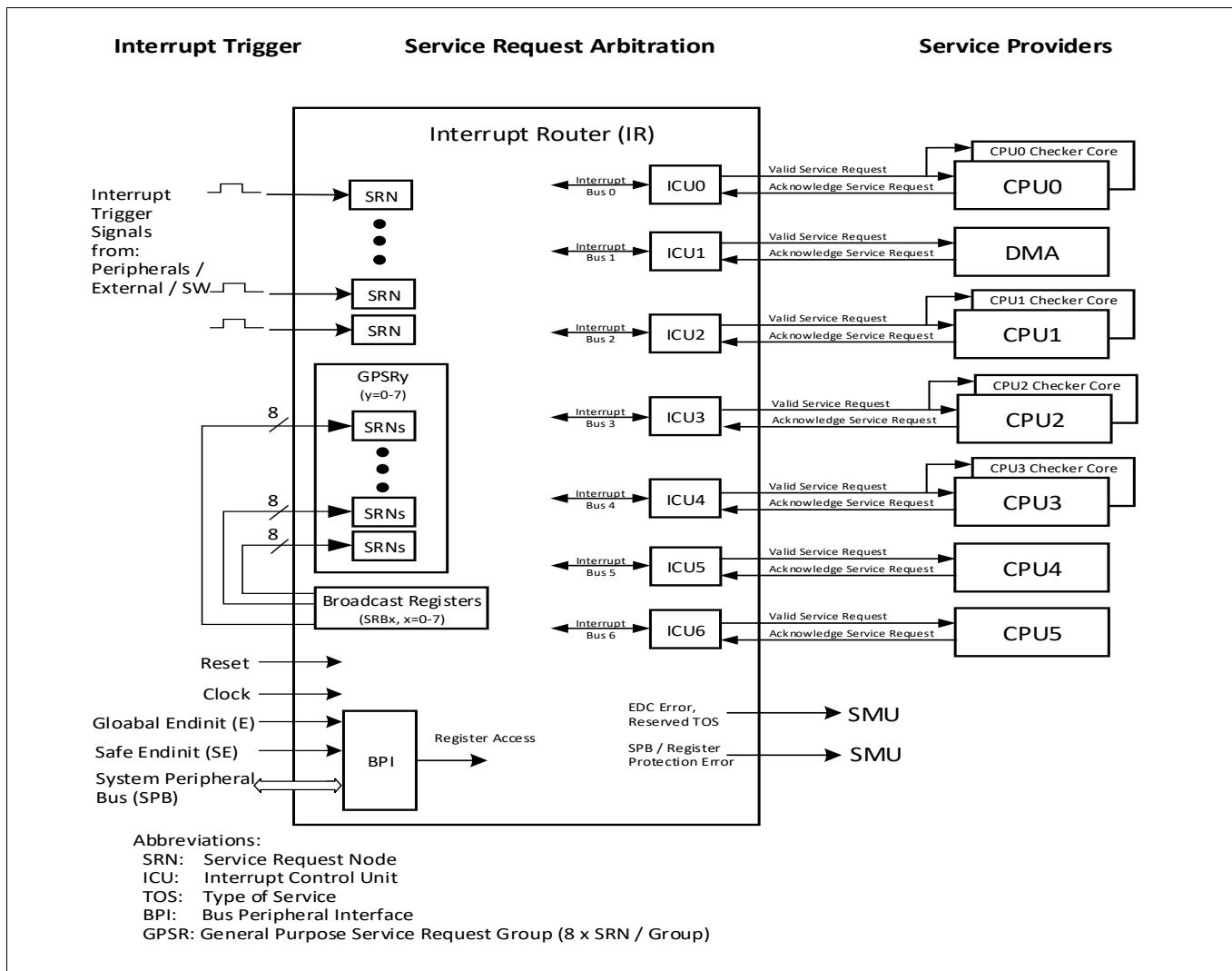


Figure 172 Block Diagram of the Interrupt System

## 16.4 Service Request Nodes (SRN)

Each Service Request node (SRN) inside the Interrupt Router module contains a Service Request Control (SRC) Register and interface logic that connects it to the triggering unit outside the Interrupt Router module and to the interrupt arbitration buses inside the Interrupt Router (see also: [Figure 172](#)).

### 16.4.1 Service Request Control Registers

All Service Request Control Registers in the Interrupt Router module have the same format. In general, these registers contain:

- Enable/disable information (SRE, [Page 9](#))
- Service Request Set bit and Service Request Clear bit (SETR, CLRR, [Page 9](#))
- Software Sticky Bit (SWS) as indication of an Software initiated Service Request (SWS, [Page 12](#))
- Service Request Priority Level vector (SRPN, [Page 10](#))
- Service Request destination / Service Provider (TOS, [Page 9](#))
- Service request status bit (SRR, [Page 9](#))
- Integrity errors signalled to the Safety Management Unit (SMU) ([Page 11](#))
- Interrupt Overflow bits (IOV, [Page 11](#))

**Interrupt Router (IR)**

Besides being activated by the associated triggering unit through hardware, each SRN can also be set or reset by software via two software-initiated service request control bits.

**16.4.1.1 General Service Request Control Register Format****16.4.1.1.1 Service Request Control Register (SRC)**

The description given in this chapter characterizes the Service Request Control registers (SRC).

Pls. note: several modules have additional interrupt related control registers on module level (e.g. interrupt status, set clear or enable register). These module registers are described in the corresponding sections of the module chapters.

**Service Request Control Register i****SRCi (i=0-1023)**

**Service Request Control Register i**      **(00000<sub>H</sub> + i\*4)**      **Debug Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SWSC LR	SWS	IOVCL R	IOV	SETR	CLRR	SRR	0						ECC	
r	w	rh	w	rh	w	w	rh	r						rwh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TOS		SRE	0						SRPN				
r		rw		rw	r						rw				

Field	Bits	Type	Description
SRPN	7:0	rw	<p><b>Service Request Priority Number</b>            The SRPN bit field defines the priority of a service request with respect to service requests with to the same service provider (same SRC.TOS configuration):  <math>00_H</math> -&gt; Service request is on lowest priority            ...  <math>FF_H</math> -&gt; Service request is on highest priority</p> <p><b>Notes</b></p> <ol style="list-style-type: none"> <li>For a CPU <math>01_H</math> is the lowest priority as <math>00_H</math> is never serviced.            For a DMA <math>00_H</math> triggers channel 0.</li> <li>For DMA, SRPN must not be greater than the highest implemented DMA channel number.</li> </ol>
SRE	10	rw	<p><b>Service Request Enable</b>  <math>0_B</math> Service request is disabled  <math>1_B</math> Service request is enabled</p>

**Interrupt Router (IR)**

Field	Bits	Type	Description
<b>TOS</b>	13:11	rw	<p><b>Type of Service Control</b>  The TOS bit field configuration maps a Service Request to an Interrupt Service Provider:</p> <p><b>Note:</b> <i>In products where DMA or CPUx is not implemented, the related TOS encoding will not be used and treated as RESERVED.</i></p> <p>3. <i>In products with less than 4 ISPs the effective size of the TOS bit field might be reduced to 1 or 2 bit.</i></p> <ul style="list-style-type: none"> <li><b>000<sub>B</sub></b> CPU0 service is initiated</li> <li><b>001<sub>B</sub></b> DMA service is initiated</li> <li><b>010<sub>B</sub></b> CPU1 service is initiated</li> <li><b>011<sub>B</sub></b> CPU2 service is initiated</li> <li><b>100<sub>B</sub></b> CPU3 service is initiated</li> <li><b>101<sub>B</sub></b> CPU4 service is initiated</li> <li><b>110<sub>B</sub></b> CPU5 service is initiated</li> <li><b>Others</b>, Reserved (no action)</li> </ul>
<b>ECC</b>	20:16	rwh	<p><b>Error Correction Code</b>  The ECC bit field will be updated by the SRN under the following conditions:</p> <ul style="list-style-type: none"> <li>• Write or Read-Modify-Write to SRC[31:0]</li> <li>• Write to SRC[15:0] (16-bit write)</li> <li>• Write to SRC[15:8] or write to SRC[7:0] (byte write)</li> </ul> <p>For more details pls. see <a href="#">Chapter 16.4.1.9</a>.</p> <p><b>Note:</b> <i>In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.</i></p>
<b>SRR</b>	24	rh	<p><b>Service Request Flag</b>  The SRR bit shows the status of the Service Request.</p> <ul style="list-style-type: none"> <li><b>0<sub>B</sub></b> No service request is pending</li> <li><b>1<sub>B</sub></b> A service request is pending</li> </ul>
<b>CLRR</b>	25	w	<p><b>Request Clear Bit</b>  The CLRR bit is required to reset <a href="#">SRR</a>.</p> <ul style="list-style-type: none"> <li><b>0<sub>B</sub></b> No action</li> <li><b>1<sub>B</sub></b> Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.</li> </ul>
<b>SETR</b>	26	w	<p><b>Request Set Bit</b>  The SETR bit is required to set <a href="#">SRR</a>.</p> <ul style="list-style-type: none"> <li><b>0<sub>B</sub></b> No action</li> <li><b>1<sub>B</sub></b> Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.</li> </ul>

**Interrupt Router (IR)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>IOV</b>	27	rh	<p><b>Interrupt Trigger Overflow Bit</b>            The IOV bit is set by HW if a new service request was triggered via interrupt trigger or <b>SETR</b> bit while the SRN has still an pending service request.</p> <p>0<sub>B</sub> No Interrupt Trigger Overflow detected            1<sub>B</sub> Interrupt Overflow Detected.</p>
<b>IOVCLR</b>	28	w	<p><b>Interrupt Trigger Overflow Clear Bit</b>            IOVCLR is required to reset <b>IOV</b>.</p> <p>0<sub>B</sub> No action            1<sub>B</sub> Clear IOV; bit value is not stored; read always returns 0.</p>
<b>SWS</b>	29	rh	<p><b>SW Sticky Bit</b>            The Software Sticky Bit is set when the <b>SRR</b> bit has been set via the <b>SETR</b> bit.            This bit can be cleared by writing with 1 to <b>SWSCLR</b>. Writing to SWS has no effect.</p> <p>0<sub>B</sub> No interrupt was initiated via SETR            1<sub>B</sub> Interrupt was initiated via SETR</p>
<b>SWSCLR</b>	30	w	<p><b>SW Sticky Clear Bit</b>            SWSCLR is required to reset <b>SWS</b>.</p> <p>0<sub>B</sub> No action            1<sub>B</sub> Clear SWS; bit value is not stored; read always returns 0.</p>
<b>0</b>	9:8, 15:14, 23:21, 31	r	<p><b>Reserved</b>            Read as 0; should be written with 0.</p>

## Interrupt Router (IR)

### 16.4.1.2 Changing the SRN configuration

All Service Request Nodes are disabled per default. To use a Service Request Node, it has to be configured and enabled by setting the SRC.SRE bit=‘1’.

The Service Request Nodes can be configured regarding the interrupt service provider target (SRC.TOS) and regarding the service request priority number (SRC.SRPN).

Once an SRN is enabled, the TOS and/or SRPN bit fields can be changed by using the following sequence:

- Disable the SRN (set SRC.SRE=‘0’)
- Check that the SRN is disabled (read back SRC.SRE and check for SRE=‘0’)
- Check the register LWSRx (Last Winning Service Request, read/poll LWSRx, see note below):
  - If LWSRx.STAT=‘0’ or LWSRx.SRPN or LWSRx.ECC are not equal to the old SRC values anymore go on with the next step (change SRC value)
  - If LWSRx.STAT=‘1’ and LWSRx.SRPN and LWSRx.ECC are equal to the (old) SRC value check LWSRx again
- Change the SRC.TOS and/or SRC.SRPN bit field
- Enable the SRN (SRC.SRE=‘1’) (write to SRC.SRE)

If the service request that has to be re-configured was just pending when it was disabled, it can be that the service request was already arbitrated and provided to the Interrupt Service Provider (CPU or DMA). In this situation the enable of the re-configured SRN should be delayed until the LWSRx poll algorithm above is fulfilled to ensure that the configuration of the just disabled SRN is not present anymore in the interrupt system. The time of the read/poll sequence is not deterministic if the disabled Service Request Node is mapped to TriCore. If the ISP is the DMA module, enter and acknowledge are signalled in the same cycle. The ‘x’ in LWSRx means that the LWSR register related to the TOS setting has to be checked (TOS=0 -> LWSR0, TOS=1 -> LWSR1, ...).

### 16.4.1.3 Protection of the SRC Registers

The SRC register is write protected via an On Chip Bus Master TAG-ID protection (see [Chapter 16.8.1](#)). This protection is controlled via the Interrupt Router control registers ACCEN\_CONFIG and ACCEN\_TOSx.

#### Access protection of SRC[31:16]

- SRC[31:16] is write protected by ACCEN\_TOSx (x = SRC.TOS)
- SRC[15:0] is write protected by ACCEN\_CONFIG

The split of the SRC write protection into different write protection registers for SRC[31:16] and SRC[15:0] allows to define a different protection configuration for the configuration of all the Service Request Nodes (that is normally static during runtime) and the Service Request Nodes control registers that might be used during runtime by the related SW task as part of normal application (e.g. software interrupt).

#### Write access to SRC[31:16] with access protection violation

When an SRC register is written with a 32 bit data access, only the SRC register parts without ACCEN protection violation will be updated:

- Violation for SRC[31:16] (ACCEN\_TOSx) and SRC[15:0] (ACCEN\_CONFIG)
- no update of the SRC registers, Alarm send to SMU
- Violation for SRC[31:16] (ACCEN\_TOSx)
- no update of SRC[31:16], updated of SRC[15:0], Alarm send to SMU
- Violation for SRC[15:0] (ACCEN\_CONFIG)
- update of SRC[31:16], no updated of SRC[15:0], Alarm send to SMU

## Interrupt Router (IR)

### Access protection of the SRC configuration (SRC[15:0])

The lower half of all SRC registers SRCx[15:0] is write protected via the ACCEN\_CONFIG register.

This means that the configuration of the ACCEN\_CONFIG defines which TAG ID is allowed to write to the lower half of all implemented SRC registers. Please note that a modification of SRC.TOS, SRC.SRE, SRC.SRPN (all mapped in SRC[15:0]) will indirectly also modify the SRC.ECC bit field.

### Background / Use Case (SRC[15:0] protected by ACCEN\_CONFIG)

SRCx[15:0] includes the configuration bits / bit fields of the Service Request Nodes that are normally:

- configured during startup
- static during runtime
- shall not be modified by non safe SW tasks

ACCEN\_CONFIG allows a configuration where only specific TAG IDs are enabled to re-configure the SRC configurations, for example a CPUx.DMI Safe TAG ID.

### Access protection of the SRC control bits (SRC[31:16])

The upper half of a SRC register SRC[31:16] is write protected via one of the ACCEN\_SRC\_TOSx registers.

The ACCEN\_SRC\_TOSx register that protects a SRN is selected by the SRC.TOS configuration of the SRN:

- SRCx.TOS = 0 -> SRC register is write protected by ACCEN\_SRC\_TOS0
- SRCx.TOS = 1 -> SRC register is write protected by ACCEN\_SRC\_TOS1
- SRCx.TOS = 2 -> SRC register is write protected by ACCEN\_SRC\_TOS2
- ...

**Note:** *For 'Reserved' TOS encodings no ACCEN\_SRC\_TOSx register is implemented. This means for an SRCy that is configured with a Reserved TOS encoding, SRCy can be always written. A service request to a not implemented ISP (SRC.TOS=Reserved) will be signalled as alarm to the SMU.*

### Background / Use Case (SRC[31:16] protected by ACCEN\_SRC\_TOSx)

SRCx[31:16] includes the control bits / bit fields of a Service Request Node that can be used during runtime to:

- Set a Service Request (SW interrupt)
- Clear a service Request
- Clear the Interrupt Overflow or Sticky bit
- Inject an ECC error via the SRC.ECC bit field

Without this mechanism, any CPU task and any on chip bus IP with master functionality could e.g. generate SW interrupts to any CPU and/or clear any interrupt.

The implemented mechanism enables the system to encapsulate SW tasks on a 'per CPU' granularity and to restrict the side effects of incorrect SW to the CPU where this SW is executed.

Example: all SRNs that are mapped to CPU0 (TOS=0) are write protected by ACCEN\_SRC\_TOS0. Means that via the ACCEN\_SRC\_TOS0 configuration it is defined which TAG ID is allowed to set or clear service requests to CPU0.

SW. ACCEN\_SRC\_TOS0 could be for example configured that only CPU0 tasks are allowed to set SW interrupts to CPU0. This ensures that a corrupted SW an any other CPU or DMA channel can not influence CPU0 by permanent Software Interrupts.

**Note:** *In this case an access protection error the write is silently ignored, an error is signaled to the SMU. Beside this signaling to the SMU, no other error, interrupt or trap is generated.*

## Interrupt Router (IR)

### 16.4.1.4 Request Set and Clear Bits (SETR, CLRR)

The SETR and CLRR bits allow software to set or clear the service request bit SRR.

- Writing a 1 to SETR causes bit SRR to be set to 1
- Writing a 1 to CLRR causes bit SRR to be cleared to 0
- Writing a 1 to SETR and CLRR at the same time, SRR is not changed
- The value written to SETR or CLRR is not stored
- Writing a 0 to these bits has no effect
- These bits always return 0 when read

### 16.4.1.5 Enable Bit (SRE)

The SRE bit enables an interrupt to take part in the arbitration for the selected service provider. It does not enable or disable the setting of the request flag SRR; the request flag can be set by hardware or by software (via SETR) independent of the state of the SRE bit. This allows service requests to be handled automatically by hardware or through software polling.

#### SRE = 1:

If SRE = 1, pending service requests are passed on to the designated service provider for interrupt arbitration. The SRR bit is automatically set to 0 by hardware when the service request is acknowledged by the Interrupt Service Provider. It is recommended that in this case, software should not modify the SRR bit to avoid unexpected behavior due to the hardware controlling this bit.

#### SRE = 0:

If SRE = 0, pending service requests are not passed on to service providers. Software can poll the SRR bit to check whether a service request is pending. To acknowledge the service request, the SRR bit must then be reset by software by writing a 1 to CLRR.

**Note:** *In this document, ‘active source’ means an SRN whose Service Request Control Register has its request enable bit SRE set to 1 to allow its service requests to participate in interrupt arbitration.*

### 16.4.1.6 Service Request Flag (SRR)

When set, the SRR flag indicates that a service request is pending. It can be set or reset directly by hardware or indirectly through software using the SETR and CLRR bits. Writing directly to this bit via software has no effect.

SRR can be set or cleared either by hardware or by software regardless of the state of the enable bit SRE. However, the request is only forwarded for service if the enable bit is set. If SRE = 1, a pending service request takes part in the interrupt arbitration of the service provider selected by the device’s TOS bit field. If SRE = 0, a pending service request is excluded from interrupt arbitrations.

SRR is automatically reset by hardware when the service request is acknowledged and serviced. Software can poll SRR to check for a pending service request. SRR must be reset by software in this case by writing a 1 to CLRR.

**Note:** *Clearing a pending service request flag SRR and enabling the corresponding service request node (SRN) should be done in two steps / two writes: first clear the SRR flag (SRC.CLRR), than enable the (SRC.SRE).*

### 16.4.1.7 Type-Of-Service Control (TOS)

Each TriCore CPU and each system DMA instance can act as an interrupt service provider (ISP).

## Interrupt Router (IR)

A Service Request Node can be mapped via the TOS bit field to exactly one ISP.

The TOS configuration maps a service request to

- an Interrupt Service Provider (CPUx, DMA)
- the Interrupt Router internal Interrupt Control Unit related to the ISP (ICUx, TOS=0 -> ICU0, TOS=1 -> ICU1, ....)
- an ACCEN\_SRC\_TOSx write protection register (TOS=0 -> ACCEN\_SRC\_TOS0, ...)

*Note:* If a SRN is enabled and configured with a Reserved TOS encoding, a trigger of this SRN by HW or SW is signalled via Alarm to the SMU, no further action.

*Note:* If a SRN is enabled and configured with a Reserved TOS encoding, a trigger of this SRN by HW or SW is signalled via Alarm to the SMU, no further action.

### 16.4.1.8 Service Request Priority Number (SRPN)

The Service Request Priority Number (SRPN) defines the priority of a service request with respect to other sources requesting service from the same service provider, and with respect to the priority of the service provider itself.

Each active SRN mapped to the same service provider (same TOS configuration):

- Can have a unique SRPN value
- Can have a non unique SRPN to give a group of Service Requests the same priority
- For a group of Service Requests with the same SRPN, the sequence of execution can not be defined
- If an SRN is not active – meaning its SRE bit is 0 – no restrictions are applied to the SRPN configuration

#### Service Provider is a CPU

Service requests are associated with Service Request Priority Numbers by an Interrupt Vector Table located in each CPU. This means that the CPU Interrupt Vector Table is ordered by priority number. This is unlike traditional interrupt CPU architectures in which their interrupt vector tables are ordered by the source of the interrupt. The CPU Interrupt Vector Tables allow a single peripheral to have multiple priorities for different purposes.

#### Notes

1. For a CPU, the SRPN value of  $0000_H$  is a special value. A Service Request with the SRPN  $0000_H$  ignored by a CPU. The CPU will not acknowledge it, the related Service Request Node will therefore not be cleared. But because priority 0 is the lowest service request priority, it will not block the Interrupt Router arbitration.
2. TriCore CPUs are providing a flexible interrupt table alignment with a configurable vector spacing of 8 byte or 32 byte. Pls. see also CPU chapter.

#### Service Provider is a DMA

Service Requests are associated with Service Request Priority Numbers to DMA channel numbers:

- SRPN=x will trigger DMA channel x, if DMA channel x implemented

Only the SRPN numbers 0 ... max\_channel\_number will result in a trigger of the related DMA channel.

SRPN numbers > max\_channel number will be handled as any other pending service request to the DMA but will not result in a DMA channel trigger inside the DMA module nor any other signalling.

#### Examples

- In case of an 16 channel DMA module the SRPN number 00H will trigger the channel 0, 07H will trigger the channel 7. All SRPN > 0FH will be handled as any other service request but will not result in a channel trigger.

## Interrupt Router (IR)

- In case of an 64 channel DMA module the SRPN number 00H will trigger the channel 0, 17H will trigger the channel 23 and 3FH. will trigger channel 64. All SRPN > 3FH will not be executed as any other service request but will not result in a channel trigger.
- A corrupted SRC configuration that is triggering a non existing DMA (due to the corruption of TOS, SRPN or SRE) will be detected by the ICU EDC check ([Chapter 16.4.1.9](#)) and signalled to the ICU as soon the service request with the corrupted configuration is acknowledged by the Interrupt Service Provider (here: DMA).

### 16.4.1.9 ECC Encoding (ECC)

The SRC.ECC bit field will be updated by the SRN under the following conditions:

- Write or Read-Modify-Write to SRC[31:0]
- Write to SRC[15:0] (16-bit write)
- Write to SRC[15:8] or write to SRC[7:0] (byte write)

In case of a 32-bit write or a Read-Modify-Write to the SRC, the ECC bit field will be updated with the calculated ECC, the data written to ECC bit field will be ignored.

ECC encoding covers the new values of SRC.SRPN, SRC.TOS, SRC.SRE and the internal 10 bit index number of the written SRN.

There is no permanent ECC check. ECC check will be checked whenever the SRN with an pending service request was accepted by the selected (TOS) service provider as next service request to be processed.

For a check of the error detection mechanisms ECC errors can be inserted (ECC bit field modified) by:

- Writing to SRC[23:16] (byte write)
- Writing to SRC[31:16] (16-bit write)

*Note: In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.*

#### ECC Code

The ECC code used for the Interrupt Router Error Detection mechanism is a Hsiao 22\_5 code with DED (double error detection) capability:

```

GEN_ENC22_5 : if (word_width_g/(nb_mems_g*ecc_granularity_g)) = 22 and
(nb_check_bits_g = 5) generate cmr22_5: for i in 0 to nb_check_bits_g - 1 generate
CODE_MATRIX_ROWS(i)      <= code_rows_22_5(i);      end generate; end generate;

type rows_22_5_t is array (4 downto 0) of std_ulogic_vector(21 downto 0); constant
code_rows_22_5 : rows_22_5_t := ("0001001011001011011011",
"00100101010101101",                      "0100100110100110110110",
"1000111000111000111111",                  "11110000001111100011"
);

```

### 16.4.1.10 Interrupt Trigger Overflow Bit (IOV)

The IOV bit is set by HW if both conditions are true:

- Service request is pending
- A new service request is triggered via interrupt trigger or SETR bit

## Interrupt Router (IR)

### 16.4.1.11 Interrupt Trigger Overflow Clear Bit (IOVCLR)

The Interrupt Trigger Overflow Clear Bit can be used to clear the IOV bit. The IOV bit is cleared by writing a '1' to the IOVCLR bit.

### 16.4.1.12 SW Sticky Bit (SWS)

The Software Sticky Bit is set when the SETR (Request Set Bit) is written with 1.

### 16.4.1.13 SW Sticky Clear Bit (SWSCLR)

The Software Sticky Clear Bit can be used to clear the SWS bit. The SWS bit is cleared by writing a '1' to the SWSCLR bit.

## 16.5 Mapping of Module Interrupt Request Triggers to SRNs

All module interrupt requests are mapped to Service Request Nodes (SRN) in the Interrupt Router.

There is one dedicated Service Request Node (SRN) for each module interrupt request.

Each SRN has one unique SRN Index Number within the Interrupt Router module. The index numbers are not required for the functionality of the interrupt router module itself. The index numbers can be used to select a Service Request Nodes for observation via the OTGM feature.

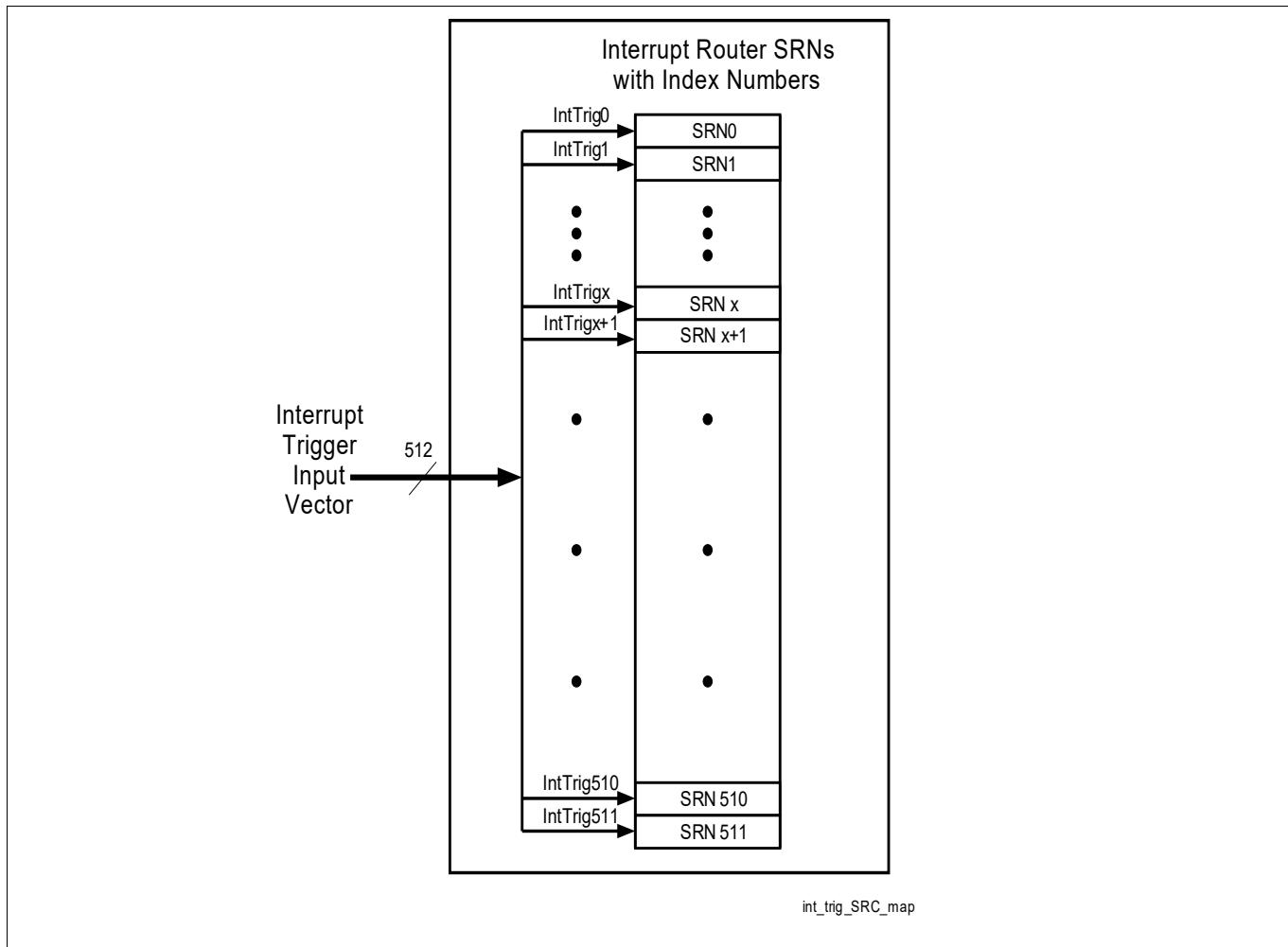
The index number of an Service Request Control (SRC) register can be directly calculated out if its address offset in the SRC address range, see also [Chapter 16.5.1](#).

The Interrupt Router module has one 1024 interrupt trigger input vector. Each interrupt trigger input vector bit [x] is related to one SRN with the SRN Index Number x<sup>1)</sup>. This means that a trigger pulse on interrupt trigger input vector bit [x] will trigger the SRN [x].

*Note:* A positive edge on a interrupt trigger signal is interpreted as interrupt trigger. See also: [Chapter 16.5.3](#).

1) The Interrupt Router register overview table shows for all module service requests the SRC registers with its address offset and the related SRN Index number

## Interrupt Router (IR)



**Figure 173 Mapping of Module Interrupt Trigger to SRN (Index Numbers)**

### 16.5.1 SRC Index Number

Each Service Request Node can be configured and controlled via its dedicated Service Request Control register (SRC). The implemented SRC registers are described in the chapter [Page 28](#).

The address of SRC registers related to one module instance is identical over the whole Aurix family (e.g. interrupts of QSPI0).

Each SRC has a unique index number inside the interrupt router module.

The index number of a Service Request Control (SRC) can be directly calculated out if its address offset in the SRC address range:

- $\text{Index}(\text{SRC}) = \langle \text{SRC Address Offset} \rangle / 4$

Example:

- SRC\_BCU\_SPB offset is 20Hex (see [Chapter 16.13](#))
- $\text{Index}(\text{SRC\_BCU\_SPB}) = 20\text{Hex} / 4 = 8$

### 16.5.2 Interrupts related to the Debug Reset

For software debug purposes the AURIX devices require some service request nodes related only to the Debug Reset. These SRNs keep its SRC register contents and the pending service request status in case of a non Debug

## Interrupt Router (IR)

Reset (e.g. an Application Reset). In combination with other debug reset related debug logic (e.g. breakpoint logic) this allows customer SW to debug situations that result in an application reset and after an application reset.

### 16.5.3 Timing characteristics of Service Request Trigger Signals

The Interrupt Router is clocked with the System Peripheral Bus (SPB) clock Rules for the module interrupt / Service Request trigger signals to the Interrupt Router:

- Trigger signals must be synchronous to SPB clock
- Interrupt Router trigger inputs are edge sensitive (positive clock edge)
- Trigger signal pulse with min. high length of one SPB clock cycle, high pulse length can be > 1 SPB clock cycle
- Debug related trigger signal pulse should be kept high by the related module until the trigger was processed

## Interrupt Router (IR)

### 16.6 Interrupt Control Unit (ICU)

The Interrupt Router module includes one ICU per service provider (CPUs and DMA modules) where each ICU is related to one service provider. SRNs can be mapped to one of the ICUs via the SRNs SRCx.TOS register bit field (see also: [Figure 172](#)).

The Interrupt Control Units (ICU):

- Manages the arbitration among competing service requests from SRNs that are mapped to the ICU
- Provides the winner of the arbitration round to the service provider
- Receives the information from the service provider which service request was accepted
- Checks the accepted service request information (ECC check)
- Signals integrity errors to the Safety Management Unit (SMU)
- Manages the clearing of acknowledged service requests in the related SRNs

**Note:** *In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.*

#### 16.6.1 ICU Interface to ISP

The interface in between the ICU and the connected ISP is covered by the IR EDC protection. In order to add robustness to this interface, ICU and ISP outputs are set to default values while no service request is signalled to the ISP and/or while the ISP does not acknowledge a service request to the ICU.

On a stuck-at-one error of the VALID signal, a CPU will ignore the default information (SRPN=8 'h 00), a DMA will trigger Channel and acknowledge the default information to the ICU where it is detected by the EDC protection and an Alarm is send to the SMU.

On a stuck-at-one error of the ACKNOWLEDGE signal, the ICU will take the default values and detect an EDC error.

As long the ICU does not send a service request to the connected ISP (VALID = '0'), the ICU output signals are set to the following default values:

- IDX = 10 'h 000
- PIPN/SRPN = 8 'h 00
- ECC = 5 'h 00

As long the ISP does not acknowledge a service request to the related ICU (ACKNOWLEDGE = '0'), the ICU output signals are set to the following default values:

- IDX = 10 'h 000
- PIPN/SRPN = 8 'h 00
- ECC = 5 'h 00

**Interrupt Router (IR)****16.6.2 ICU Control Registers**

This section describes the Interrupt Control Unit (ICU) registers. Each ICU includes two control registers:

- Latest Winning Service Request register ([Page 16](#)) provides information about the winner of the last service request arbitration round
- Last Acknowledged Service Request register ([Page 17](#)) provides information about the last service request that was accepted by the service provider
- Error Capture register ([Page 17](#)) captures the Last Acknowledged Service Request ([Page 17](#)) register contents when an ECC error was detected by the ICU

**16.6.2.1 Latest Winning Service Request Register (LWSR)****Latest Winning Service Request Register x, related to ICUx**

The Latest Winning Service Request register provides informations about the winner of the last arbitration round. The register bit fields are representing what is provided by the ICU to the Interrupt Service Provider.

**LWSRx (x=0-7)**

**Latest Winning Service Request Register x, related to ICUx(0200<sub>H</sub>+x\*10<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

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
ST															0			ECC			0									PN	

Field	Bits	Type	Description
PN	7:0	r	<b>Latest Winner Priority Number</b> This bit field shows the Priority Number of a pending service request that won the last arbitration round. This bit field is only valid if STAT is set to 1
ECC	14:10	r	<b>Latest Winner ECC</b> This bit field shows the ECC field (SRN.ECC) that was transferred from the last winning SRN to the ICU. This bit field is only valid if STAT is set to 1.  Note: <i>In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.</i>
ID	25:16	r	<b>Latest Winner Index Number</b> This bit field shows the ID number of the last winning SRN. This bit field is only valid if STAT is set to 1
STAT	31	r	<b>LWSR Register Status</b> The STAT register indicates if the PN, ECC and ID bit fields are still valid. They are still valid if the interrupt from the SRN identified by ID is still pending. If the ICU does not have an winner because no interrupt is pending or not yet arbitrated then it clears the STAT bit. 0 <sub>B</sub> LWSR bit fields are not valid 1 <sub>B</sub> LWSR bit fields are valid

## Interrupt Router (IR)

Field	Bits	Type	Description
<b>0</b>	9:8, 15, 30:26	r	<b>Reserved</b> Read as 0; should be written with 0.

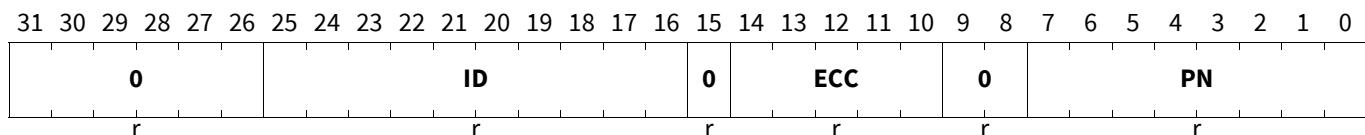
### 16.6.2.2 Last Acknowledged Service Request Register (LASR)

#### Last Acknowledged Service Request Register x, related to ICUx

The Last Acknowledged Service Request register is representing the informations about the last service request that was acknowledged by the Interrupt Service Provider. The register bit fields show what was sent by the Interrupt Service Provider together with the latest acknowledge.

#### LASRx (x=0-7)

**Last Acknowledged Service Request Register x, related to ICUx(0204<sub>H</sub>+x\*10<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>PN</b>	7:0	r	<b>Last Acknowledged Service Request Priority Number</b> This bit field shows the Priority Number of the last acknowledged service request
<b>ECC</b>	14:10	r	<b>Last Acknowledged Interrupt ECC</b> This bit field shows the ECC value of the last acknowledged service request, as send by the service provider with acknowledge  Note: <i>In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.</i>
<b>ID</b>	25:16	r	<b>Last Acknowledged Interrupt SRN ID</b> This bit field shows the ID number of the last acknowledged service request, as sent by the service provider with acknowledge
<b>0</b>	9:8, 15, 31:26	r	<b>Reserved</b> Read as 0; should be written with 0.

### 16.6.2.3 Error Capture Register (ECR)

#### Error Capture Register x, related to ICUx

On detection of an ECC error, the Error Capture register (**ECRx**) captures the related Service Request information. This is done by updating the ECR with the content of the Last Acknowledged Service Request (**LASRx**) register on detection of the ECC error. The ECR shows always the last ECR contents where an ECC error was detected. Software can clear the ECR bitfields PN, ECC, ID by writing to the **ECRx**. Error Status (STAT) and Error Overflow (EOV) bits can be used as error handling mechanism and indication that error information were lost.

**Interrupt Router (IR)**

If ECR.EOF is cleared by SW, ECR.EOF must be cleared together with ECR.STAT. If a new error is detected in parallel to the ECR.EOF clear than ECR.EOF is set again by hardware while ECR.STAT is cleared.

**Note:** *In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.*

**ECRx (x=0-7)****Error Capture Register x, related to ICUx (0208<sub>H</sub>+x\*10<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

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
<b>ST AT</b>	<b>EO V</b>	<b>ST AT</b>	<b>EO VC</b>	<b>0</b>																			<b>ECC</b>	<b>0</b>				<b>PN</b>			

Field	Bits	Type	Description
<b>PN</b>	7:0	rwh	<b>Service Request Priority Number</b> This bit field shows the priority number of the last service request where an error was detected. Bit field can be modified by writing to it.
<b>ECC</b>	14:10	rwh	<b>Service Request ECC</b> This bit field shows the ECC of the last service request where an error was detected. Bit field can be modified by writing to it. This bit field can be modified by: <ul style="list-style-type: none"><li>• Writing to SRC[23:16] (byte write)</li><li>• Writing to SRC[31:16] (16-bit write)</li></ul>
<b>ID</b>	25:16	rwh	<b>Service Request Node ID</b> This bit field shows the ID of the last service request where an error was detected. Bit field can be modified by writing to it
<b>EOVCLR</b>	28	w	<b>Error Overflow Bit</b> The EOVCLR bit is used to clear the EOF bit. The EOF bit must be cleared together with the STAT bit. $0_B$ No action $1_B$ Clear EOF bit; bit value is not stored; read always returns 0; no action if EOF bit is set in parallel.
<b>STATCLR</b>	29	w	<b>Error Status Bit</b> The STATCLR bit is used to clear the STAT bit. $0_B$ No action $1_B$ Clear STAT bit; bit value is not stored; read always returns 0; no action if STAT bit is set in parallel.
<b>EOF</b>	30	rh	<b>Error Overflow Bit</b> The bit is set if an ECC error was detected by the ICU while ECR.STAT='1' (Error Overflow situation). $0_B$ No Error Overflow situation detected $1_B$ Error Overflow situation detected

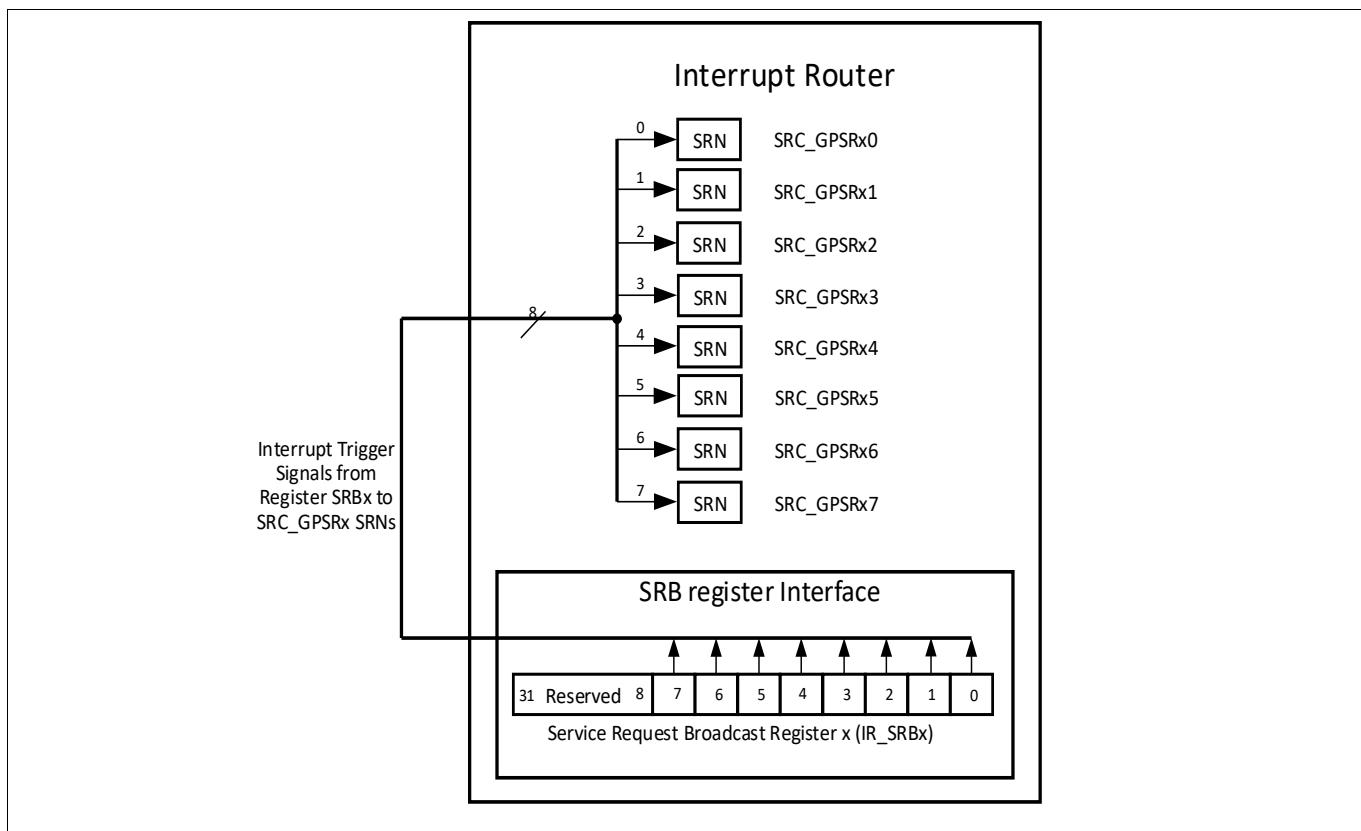
## Interrupt Router (IR)

Field	Bits	Type	Description
<b>STAT</b>	31	rh	<b>Error Status Bit</b> The Error Status Bit is set whenever an ECC was detected by the ICU. $0_B$ No ECC error detected $1_B$ ECC error detected
<b>0</b>	9:8, 15, 27:26	r	<b>Reserved</b> Read as 0; should be written with 0.

## 16.7 General Purpose Service Requests, Service Request Broadcast

The INT module provides multiple groups of General Purpose Service Requests (GPSR) and a mechanism to trigger multiple Service Requests of a GPSR group in parallel, by software ([Chapter 16.7.2](#)).

General Purpose Service Requests are intended for Software Interrupts because they are not mapped to a hardware interrupt trigger events.



**Figure 174 Structure of a General Purpose Service Request Group and the related Broadcast register (Example for SRC\_GPSR0x, SRBx).**

### 16.7.1 General Purpose Service Requests (GPSRxy)

The INT module provides multiple groups of General Purpose Service Requests:

- Each General Purpose Service Request Group consists of eight Service Request Nodes.
- The General Purpose Service Requests are named SRC\_GPSRxy<sup>1)</sup>
- The GPSR are intended for SW interrupts (not mapped to hardware service request triggers)

1) SRC\_GPSRxy: x = group number; y= number of interrupt within the group, y=0:7

## Interrupt Router (IR)

- A GPSR can only be triggered by writing '1' to the related SRC\_GPSRxy.SETR<sup>1)</sup> bit or by writing a '1' to the related Service Request Broadcast register bit in SRBx[y])

### 16.7.2 Service Request Broadcast Registers (SRBx)

Service Broadcast registers (SRBx) can be used to set service requests to multiple Service Providers (CPU or DMA) in parallel.

There is one Service Request Broadcast register (SRBx) implemented for each General Purpose Service Request Group (GPSRxy<sup>1</sup>).

A Service Request Broadcast register (SRBx) can be used to trigger multiple Service Requests within the SRC\_GPSR<sup>1</sup> group in parallel (see [Figure 174](#)).

- A Service Request Broadcast Register is always read as 0
- Writing '1' to SRBx[y] triggers the service request GPXRxy<sup>1)</sup>
- Writing '1' to SRBx[31:6] has no effect.

### 16.7.3 Access protection of SRBx registers (ACCEN\_SRBx)

Each SRBx register is write protected via a dedicated ACCEN\_SR<sub>x</sub>0 / ACCEN\_SR<sub>Bx</sub>1 register set:

- Each SRBx register has a related ACCEN\_SRBx register (x = same number)
- The configuration of the ACCEN\_SRBx defines which TAG ID is allowed to write to the related SRBx register
- In the case of an access protection violation the write is silently ignored, an error is signaled to the SMU. Beside this signaling to the SMU, no other error, interrupt or trap is generated.

Use case: ACCEN\_SRBx defines which TAG ID is allowed to write to SRBx in order to trigger multiple Service Request Nodes of the General Purpose Service Request Group x.

## 16.8 System Registers

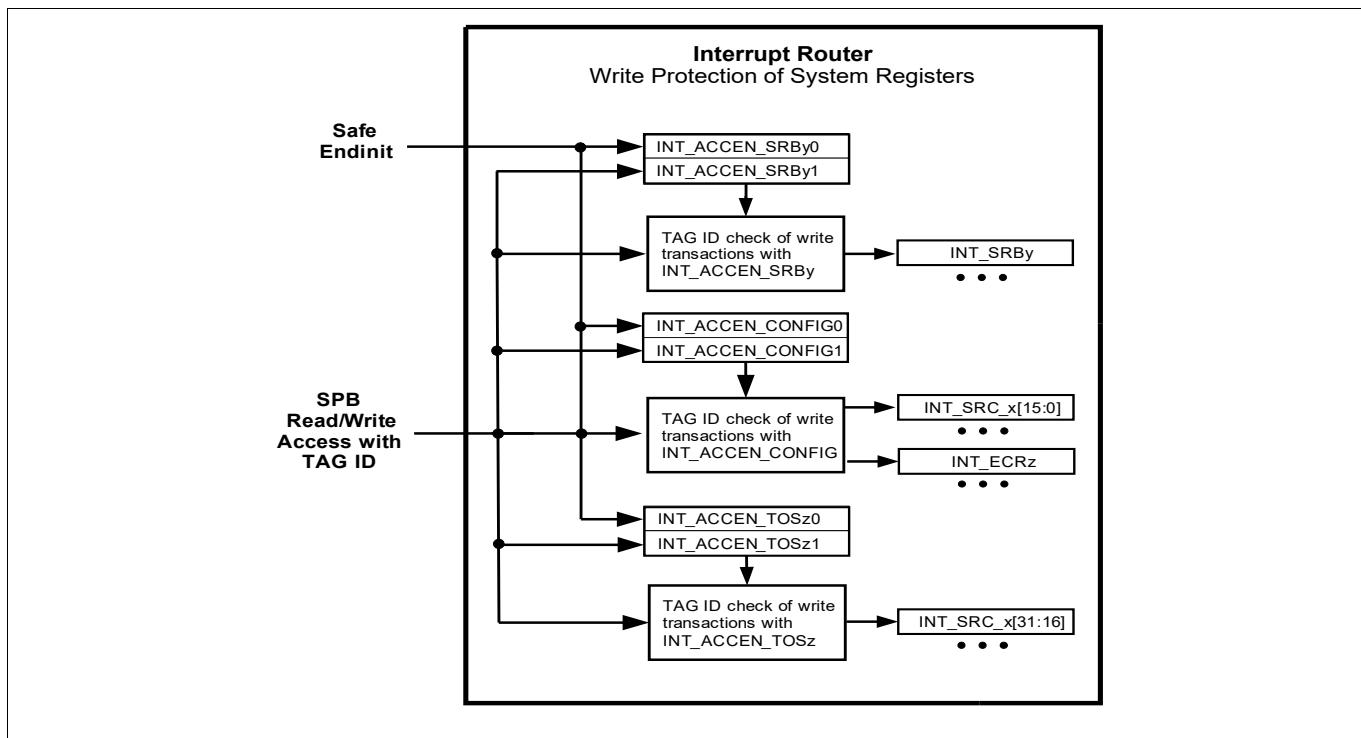
The Interrupt Router module does not support the CLC, the OCS and the KRSTx registers. The interrupt router supports multiple sets of access enable (ACCEN) registers:

- INT\_ACCENx: Register access protection is implemented with the standard ACCEN register but with the registers ACCEN\_CONFIG0/1, ACCEN\_SR<sub>Bx</sub>0/1 and the ACCEN\_SRC\_TOS<sub>x</sub>0/1
- INT\_CLC: the Interrupt Router module does not support this Clock Control (CLC) feature
- INT\_KRSTx: the Interrupt Router module does not support the Module Kernel Reset feature
- OCS: the interrupt router does not support the OCDS Control and Status Register

### 16.8.1 Write Protection of Interrupt Router registers

The Interrupt Router module provides a master TAG ID based write access protection as part of the AURIX safety concept. Each on chip resource with direct or indirect bus master capability has a unique master TAG ID that can be used to identify the master of an on chip bus transaction (see also chapter On Chip Bus Systems).

## Interrupt Router (IR)



**Figure 175 Write Protection of Interrupt Router Registers**

The Interrupt Router module provides write protection to the control registers via three Access Enable Registers / Register Sets (see [Figure 175](#)):

### Access protection of the Service Request Broadcast Registers (ACCEN\_SRBy0/1)

The interrupt router module provides one or more groups of General Purpose Service Request groups, GPSSxy:

- x is the number of the General Purpose Service Request Group (GPSRx)
- y is the number of the Service Request Node within the Group (y=5:0, one group has 6 SRNs)

For each GPSRx group a dedicated Service Request Broadcast Register (SRBy) is implemented that can be used to set multiple Service Request Nodes of the group in parallel.

For more details pls. see [Chapter 16.7.3](#)

### Access protection of the static control registers (ACCEN\_CONFIG0/1)

The ACCEN\_CONFIG provides write access protection for the following registers:

- SRCx[15:0], lower 15 bit of all SRC registers. This includes the SRC bit fields Type of Service (TOS), Service Request Enable (SRE) and Service Request Priority Number (SRPN) (For more details pls. see [Chapter 16.4.1.3](#))
- ECRx, all ICU Error Capture Registers (ECR).

### Access protection of the SRC control registers (ACCEN\_SRC\_TOSy)

The ACCEN\_SRC\_TOSy provides write access protection for the following registers:

- SRCx[31:16]

For more details pls. see [Chapter 16.4.1.3](#)

## Interrupt Router (IR)

### 16.8.2 Kernel Reset Registers (KRST1/0, KRSTCLR)

The INT module does not include the kernel reset registers (KRST1, KRST0, KRSTCLR).

**Note:** *The Interrupt Router module does not support a module kernel reset.*

### 16.8.3 Clock Control Register (CLC)

The INT module does not include the module clock control (CLC).

**Note:** *The Interrupt Router module does not support the Clock Control register functionality which means that the Interrupt Router module clock can not be disabled by the CLC register.*

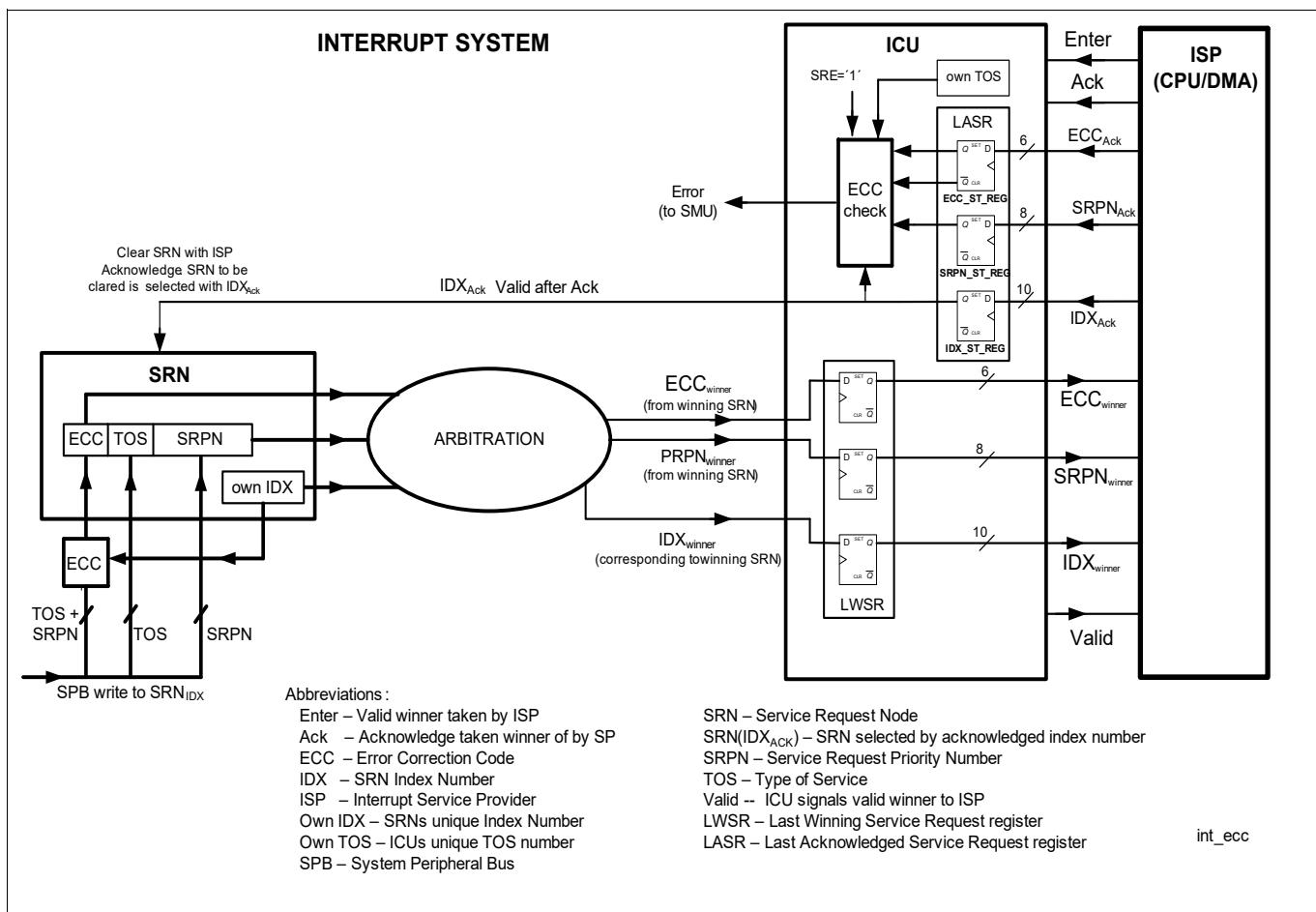
### 16.8.4 OCDS Control and Status Register (OCS)

The INT module does not include OCDS Control and Status (OCS) register.

**Note:** *The Interrupt Router module does not support the OCS register functionality.*

## 16.9 Arbitration Process

Each ICU in the interrupt module has its own interrupt bus. Each Service Request Node (SRN) can be directed to one service provider by mapping it via the SRC.TOS bit field setting to the related ICU / Interrupt Bus.



**Figure 176 Interrupt System Arbitration Scheme Overview**

## Interrupt Router (IR)

With a first pending service request, the related interrupt bus is starting a first arbitration process. The related Interrupt Control Unit provides the service request that won the last arbitration process.

The arbitration process uses 3-4 system peripheral bus clock cycles to determine the pending service request with the highest priority number, SRPN. The exact number of the implementation is described in the Module Implementation chapter.

In an arbitration process, the interrupt bus compares the SRC.SRPN bit fields of all pending Service Request Nodes, mapped to this interrupt bus via SRC.TOS setting. During the arbitration process, the pending service request with the highest priority number is identified as winner and the related SRN Service Request Control register bit field values SRPN, ECC and the Index of the SRN are provided to the ICU. The ICU provides these (SRPN, ECC, SRN Index) to the service provider. The ICU does an ECC check when it gets these information back from the service provider with an acknowledge. The ECC check is done with the received values: ECC, SRPN, SRN Index Number, SRE bit assumed to be '1' (SRN enabled) and the TOS number of the ICU.

The Interrupt Router module signals detected errors to the Safety Management Unit (one bit in the SMU, covers errors from all SRN and ICUs).

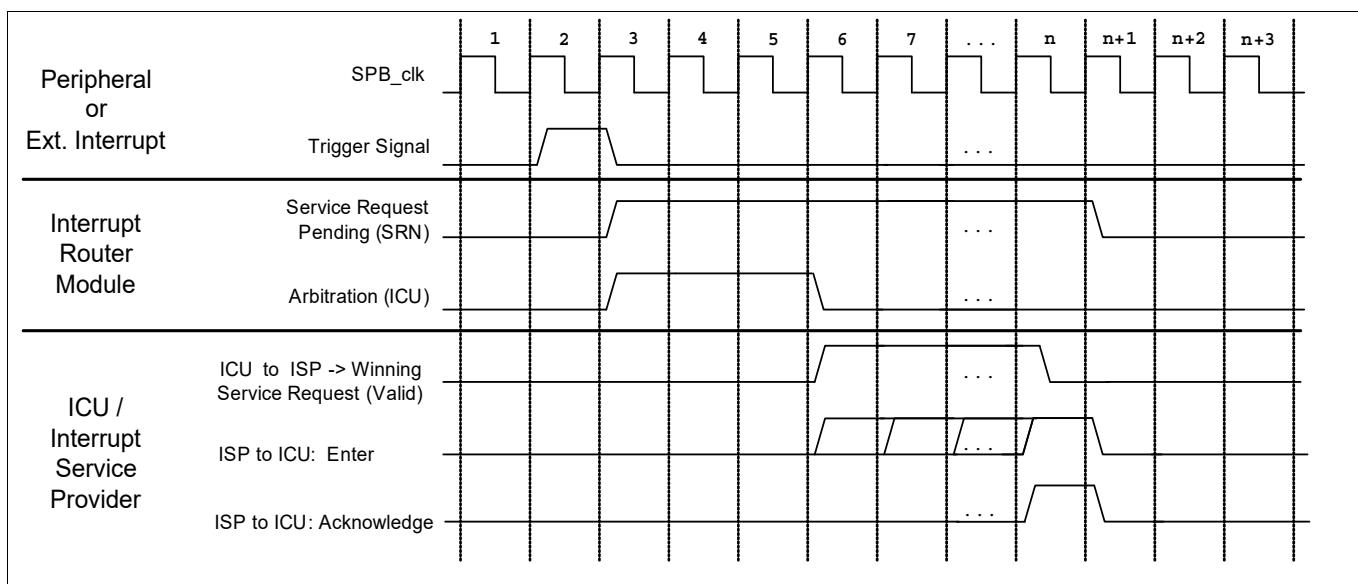
**Note:** *In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.*

### 16.9.1 Number of Clock Cycles per Arbitration Process

The Interrupt Router implementation can be configured regarding the number of:

- Supported service requests (Service Request Nodes, SRN, up to 1024)
- SRPN bit size (8-bit)
- Supported service providers (Interrupt Control Units, ICUs)
- Clock cycles per service request arbitration (3-4 SPB clock cycles)

The characteristics of the Interrupt Router module implemented in an AURIX product is described in the Module Implementation sub-chapter.



**Figure 177 Interrupt System Timing (Schematic Overview, 3 Cycle Arbitration)**

**Figure 177** shows the interrupt timing of an Interrupt Router implementation with 3 cycle arbitration.

Cycle 1: No service request for the ICU is pending (therefore no arbitration round)

## Interrupt Router (IR)

Cycle 2: One module is triggering a service request by sending a pulse to the related SRN in the Interrupt Router module.

Cycle 3-5: Arbitration among all pending service requests to the ICU

Cycle 6: ICU provides winning service request to the service provider (SRPN, ECC, SRN Index)

Cycle 7 - n-1: ICU re-arbitrates whenever a new service request from another SRN is pending, provides new winning service request to ICU if there is a new pending one with a higher SRPN number (higher priority)

Cycle 6- n-1: Interrupt Service Provider takes the information of the latest winning SRN (Enter)

Cycle n: Service provider acknowledges service request (provides SRPN, ECC, SRN Index informations of the acknowledged service request). ICU changes signals to ICU to 'no valid service request available' in the same clock cycle.

Cycle n+1: ICU does an ECC check of the acknowledge information (SRPN, ECC, SRN Index, SRE='1', TOS number of the ICU). If mismatch -> Integrity Error signalled to SMU/ SRPN, ECC and Index are captured in the ECR). The SRN with the acknowledged Index Number is cleared by hardware.

Cycle n+2: If at least one service request to the ICU is pending: new arbitration among all pending service request to the ICU

### 16.9.2 Service Request Valid

The ICU signals the ISP the information of the pending Service Request that won the arbitration round by asserting the Valid signal. Re-arbitration will be done until the ISP asserts the Enter signal.

### 16.9.3 Service Request Enter

The Interrupt Service Provider signals to the ICU that it took over the Valid information of a Service Request and starts now to prepare itself for the execution of the related Interrupt Service Routine. The Enter signal can re-set by the ISP either with Acknowledge (the Interrupt Service Routine has started, ICU will clear the related SRN) or without Acknowledge (e.g. CPU got a trap or exception before it was able to start the ISR, ICU will not clear the related SRN).

### 16.9.4 Service Request Acknowledge

When a Service Provider starts with the execution of a service request that was provided by the ICU, the Service Provider sends an acknowledge to the ICU. In parallel to the acknowledge, the Service Provider sends the informations about the executed service request back to the ICU (SRPN, ECC, SRN Index Number).

In the same clock cycle, the Service Provider sends an acknowledge, the ICU changes to 'no service request available': the ICU does not provide an arbitration winner to the service provider. This behavior of the ICU ensures that a just acknowledged service request is not provided again before the SRN was re-set (see [Figure 177](#)).

### 16.9.5 Handling of detected ECC Errors

The ICU does an Error Detection check of the informations it receives with the acknowledge from the Service Provider and the TOS number of the ICU itself. Assumption for the SRE bit is '1' (SRN was enabled). ECC in the SRN is covering the SRC bit field values: SRC.SRPN, SRC.SRN Index, SRC.SRE and SRC.TOS. (see [Figure 177](#))

Whenever the ICU receives an Acknowledge from the Interrupt Service Provider it does an Error Detection check. The check is done on the informations received with the acknowledge from the Service Provider (TOS, SRPN, Index and ECC), the TOS number of the ICUX itself and an SRE bit assumed as '1' (SRN was enabled). The ECC in the SRN is covering the SRC bit field values: SRC.SRPN, SRC.SRN Index, SRC.SRE and SRC.TOS. (see [Figure 177](#))

## Interrupt Router (IR)

**Note:** *In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.*

If the ICUs detect an ECC error:

- ICUs captures the ECC, SRPN and the Index that showed an ECC error in the Error capture registers (ECRx), sets the ECRx.STAT bit and the ECRx.EOF bit if the STAT bit is still set.
- ICU signals the error via the Interrupt Router error signal to the AURIX™ TC3xx Platform Safety Management Unit (SMU). The SMU forwards this information to a CPU (if enabled).
- ICUs clears the service request in the SRN (selected by the Index)
- The CPU that received the ‘ECC error detected in Interrupt Router’ information via SMU can identify the ICU via the Error Status bits (ECRx.STAT=‘1’), read out the related service request information and clear the ECRx.STAT bit by writing with ‘1’. The CPU can also identify if one or multiple ECC errors where detected by this ICU via the Error Overflow bit (ECRx.EOF=‘1’).

## 16.10 Usage of the Interrupt System

The following sections provide a short description of the Service Provider interfaces to the Interrupt Router ICUs.

**Note:** *All ICU sub-modules in the Interrupt Router have the same functionality.*

### 16.10.1 CPU to ICU Interface

The interrupt router module has a dedicated Interrupt Control Unit (ICU) for each CPU and DMA module. The CPU ICU interface consists of a register set where the CPU takes over the informations of a service request provided by the ICU (SRPN, SRN Index, ECC). The informations will be send back to the ICU when the CPU acknowledges the provided service request (see also [Chapter 16.4.1.8](#)).

The CPU ICU interface contains an Interrupt Control Register (ICR) that holds the current CPU priority number (CCPN), the global interrupt enable/disable bit (IE) and the pending interrupt priority number (PIPN). Further details of the CPU ICU interface and the CPU handling of interrupts can be found in the CPU chapter.

### 16.10.2 DMA to ICU Interface

The interrupt router module has dedicated Interrupt Control Unit (ICU) for each DMA module.

The DMA takes over the service request informations from the ICU, triggers the related DMA channel and acknowledge it immediately to the ICU where the related SRN is cleared.

The DMA to ICU interface consists of a register set where it takes over the information of a service request provided by the ICU (SRPN, SRN Index, ECC). The DMA sends them back to the ICU in the next clock cycle with an acknowledge (see also [Chapter 16.4.1.8](#)).

The DMA channel priority scheme is identical to the SRPN priority scheme:

- Lowest priority within the DMA: channel 0
- Lowest priority within the Interrupt Router: SRPN = 0

### 16.10.3 Software-Initiated Interrupts

Any Service Request Node can be used Software interrupt. Software can set the service request bit (SRR) in any SRN by writing to its Service Request Control Register. Thus, software can initiate service requests that are handled by the same mechanism as hardware initiated service requests.

## Interrupt Router (IR)

After the SRR bit is set in an SRN, there is no way to distinguish between a software initiated service request and a hardware initiated service request. For this reason, software should only use SRNs and interrupt priority numbers that are not being used for hardware initiated service requests.

The device contains groups of General Purpose Service Request SRNs per CPU that support software-initiated interrupts. One group per implemented TriCore CPU, each group including 8 SRNs. These SRNs are not connected to internal or external hardware trigger signals and can only be used as software interrupts / software initiated service requests. These SRNs are called General Purpose Service Requests Nodes (SRC\_GPSR $xy$ ,  $x$ =group number,  $y$ =0-7).

Additionally, any otherwise unused SRN can be employed to generate software interrupts.

### 16.10.4 External Interrupts

Eight SRNs (Int\_SCUSRC[7:0]) are reserved to handle external interrupts. The setup for external GPIO port input signals (edge/level triggering, gating etc.) that are able to generate an interrupt request is controlled in the External Request Unit (ERU). The ERU functionality is described in detail in the SCU chapter.

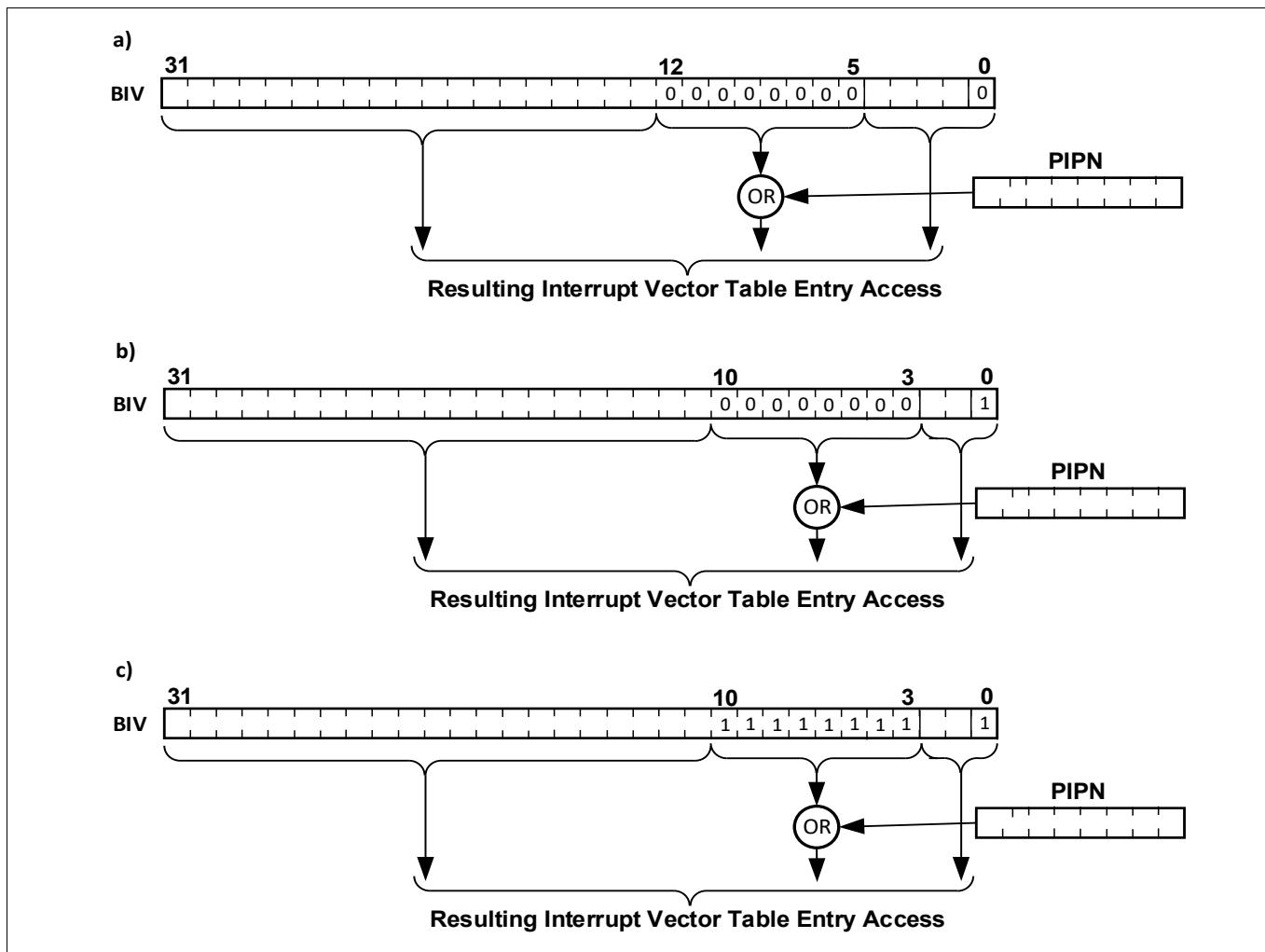
## 16.11 Use Case Examples

This section shows a use case for the interrupt system and a use case for the OTGS.

### 16.11.1 Use Case Example Interrupt Handler

This section explains how to organize the TriCore interrupt vector table. When an interrupt is accepted by the TriCore, the entry address into the interrupt vector table is calculated by the base interrupt vector table pointer TriCore register BIV and the priority number of that interrupt (PIPN). The TriCore TC1.6P and TC1.6E architecture offers the possibility to configure the vector spacing per entry to either 32 Byte (see Figure below a ([Figure 178](#)) or 8 Byte (see Figure below b ([Figure 178](#))). As a third option the vector table can be reduced to a single entry by masking the PIPN (see Figure below c ([Figure 178](#))).

## Interrupt Router (IR)



**Figure 178 Interrupt Vector Table Address Calculation for a) 32 Byte b) 8 Byte vector entry or c) single entry**

By using the 32 Byte configuration small interrupt routines can be implemented directly into the vector table. They can even span multiple vector entries (see also TriCore Architecture Manual). This type of fast interrupt handling is useful, if the vector table can be located into the TriCore program side memory. The 8 Byte configuration reduces the vector table size. Each vector entry contains only a jump instruction or a call and return as 16-bit op-code instruction. The TriCore compiler supports this kind of interrupt vector table generation by keywords or functions. A minimum vector table can be configured if the BIV mask the PIPN so that any interrupt address calculation results in the same address. E.g.

```
_mtcr(BIV,0x80000001 | 0xFF<<3); // move to core register BIV
```

This configures the BIV register to use a common, single entry where a function interruptHandler is located to branch to the specific interrupt routine by using an array of function pointers. If a pointer to the array is used the array could be switched quickly.

### Step description to initialize and install interrupts

- (Line 1) define ISR pointer array. Max. 255 interrupts possible.
- (Line 2) define pointer which points to the start of the isr\_pointer\_array.
- (Line 3) start of function interruptHandlerInstall. This function installs the interrupts in the array. Necessary information are the interrupt priority and ISR entry address.
- (Line 4) This line stores the ISR entry address in the array.

## Interrupt Router (IR)

(Line 5 and 6) This function branches to the specific interrupt routine and gets called immediately after an interrupt occurs.

(Line 7) This line gives the return command, after the ISR has been processed.

### C Code Example to initialize and install interrupts

```
(1) void (*isr_pointer_array[256])(void); (2) void (**isr)(void) = isr_pointer_array;
(3) void interruptHandlerInstall(long int SRprio, long int addr){ (4)
*isr_pointer_array[SRprio]=addr; } (5) void interruptHandler(void){ (6) isr[__mfcr(ICR)
& 0xFF](); (7) asm (" rfe"); // return from event }
```

The interrupt entry addresses are stored in a data array instead of encoding the values into the instructions. The function interruptHandlerInstall organizes the installation of the interrupts in that array (see the application of this interrupt handler in a module e.g. use case example in STM chapter). This kind of vector table generation offers sometimes more flexibility than the 8 Byte configuration and does not require any specific compiler support for interrupts.

**Note:** Before an interrupt is able to occur, the interrupt system has to be globally enabled. The Interrupt Control Register (ICR) holds the global interrupt enable bit (ICR.IE) which enables the CPU service request system. Most compiler support the attribute (or similar):  
`_enable();`  
 to set this bit. (See also Architecture Manual for more details)

## 16.12 Module Implementation

### 16.12.1 Characteristics of the Interrupt Router Module

**Table 556** shows the Interrupt Router configuration as implemented in the different TC3xx devices.

Shared arbitration is implemented per ICUx pair, starting with ICU0 (ICU0/ICU1, ICU2/ICU3, ....)

**Table 556 Mapping of Interrupt Service Provider to ICUs and implemented SRNs**

	CPU0	DMA	CPU1	CPU2	CPU3	CPU4	CPU5
TC39x	ICU0	ICU1	ICU2	ICU3	ICU4	ICU5	ICU6
TC38x	ICU0	ICU1	ICU2	ICU3	ICU4	-	-
TC37x ED / PD	ICU0	ICU1	ICU2	ICU3	-	-	-
TC36x	ICU0	ICU1	ICU2	-	-	-	-
TC33x ED	ICU0	ICU1	ICU2	-	-	-	-
TC33x PD	ICU0	ICU1	-	-	-	-	-
TC35x	ICU0	ICU1	ICU2	ICU3	-	-	-

## 16.13 Interrupt Router System and Module Registers

**Figure 179** shows all registers associated with the Interrupt Router module in the device. The Interrupt Router allocates two address ranges:

- 2 \* 256 byte address range covering the Interrupt Router system registers, ICU control registers and OTGM registers
- 8 KByte address range covering the Service Request Control registers

## Interrupt Router (IR)

**Table 557 Register Address Space - INT**

Module	Base Address	End Address	Note
INT	F0037000 <sub>H</sub>	F0037FFF <sub>H</sub>	IR Status and Control Registers

**Table 558 Register Address Space - SRC**

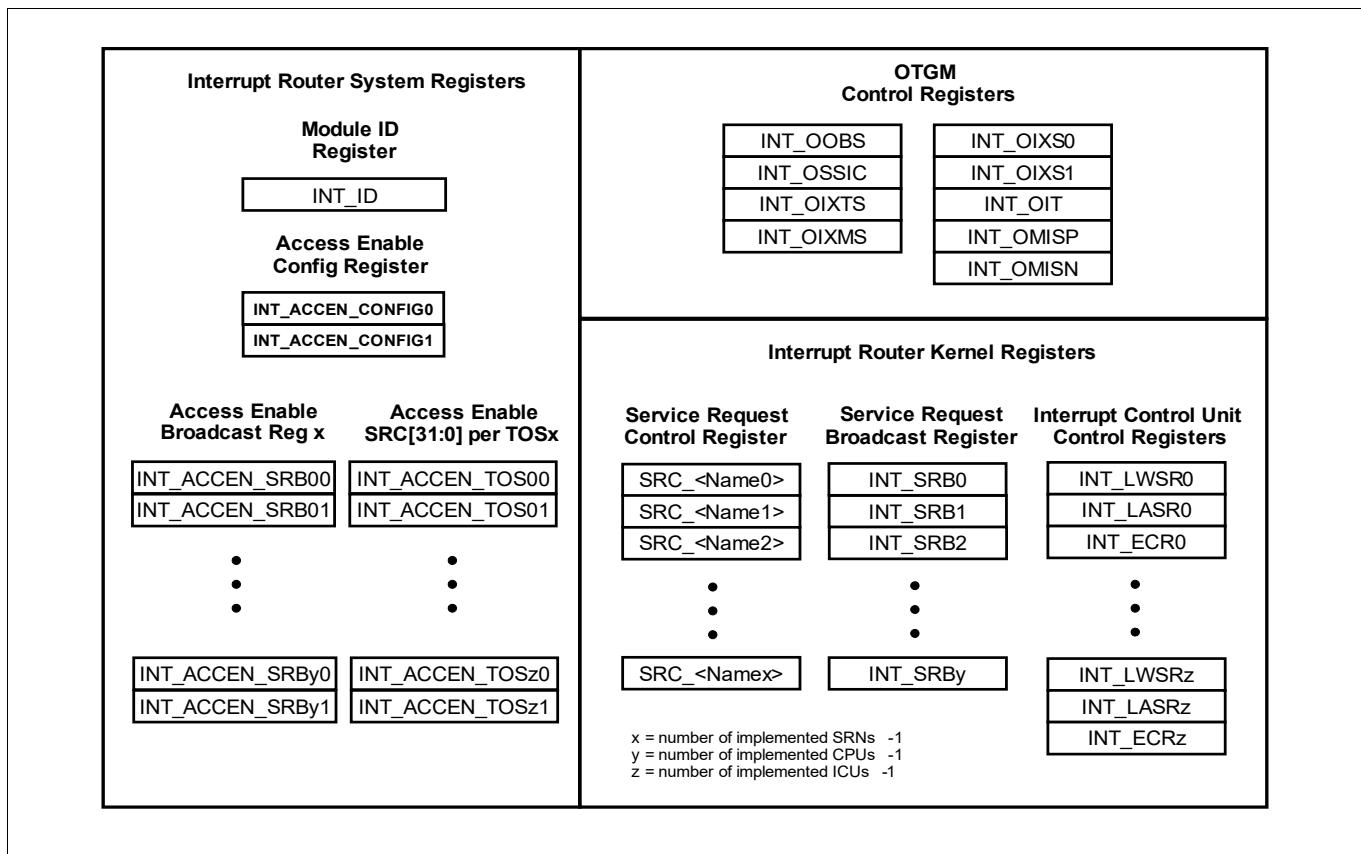
Module	Base Address	End Address	Note
SRC	F0038000 <sub>H</sub>	F0039FFF <sub>H</sub>	IR Service Request Control Registers (SRC)

### List of used Access Protection Register abbreviations

- P0 -> ACCEN\_SRBy, write protection of the related SRBy register. Number of Service Request Broadcast registers (SRB) and the related ACCEN\_SRBy registers is equal to the number of implemented TriCore CPUs.
- P1 -> ACCEN\_CONFIG, write protection of all SRCx[15:0] and ICUx Error Capture registers (ECRx)
- P2 -> ACCEN\_SRC\_TOSx, write protects bits [31:16] of all SRCs that are mapped to TOSx (SCR.TOS=x). For each implemented Interrupt Control Unit, one ACCEN\_SRC\_TOSx register is implemented.

**Note:** A violation of the access protection will not be executed (e.g. a write to a 'Px'/ACCEN protected register by an SPB access with a disabled Master TAG-ID). In this case an access protection error is signaled to the SMU. Beside this signaling to the SMU, no other error, interrupt or trap is generated.

## Interrupt Router Module Registers



**Figure 179 Interrupt Router module registers**

**Interrupt Router (IR)****Table 559 Register Overview - INT (sorted by Name)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Page Number</b>
ACCEN_CONFIG0	Access Enable covering all INT_ECRx and all SRCy[15:0], Register 0	00F0 <sub>H</sub>	<a href="#">32</a>
ACCEN_CONFIG1	Access Enable covering all INT_ECRx and all SRCy[15:0], Register 1	00F4 <sub>H</sub>	<a href="#">32</a>
ACCEN_SRBy0	Access Enable covering SRBy, Register 0	0100 <sub>H</sub> +x*8	<a href="#">32</a>
ACCEN_SRBy1	Access Enable covering SRBy, Register 1	0104 <sub>H</sub> +x*8	<a href="#">33</a>
ACCEN_SRC_TOSy0	Access Enable covering all SRCx[31:16] mapped to ICUx, Register 0	0180 <sub>H</sub> +x*8	<a href="#">33</a>
ACCEN_SRC_TOSy1	Access Enable covering all SRCx[31:16] mapped to ICUx, Register 1	0184 <sub>H</sub> +x*8	<a href="#">34</a>
ECRx	Error Capture Register x, related to ICUx	0208 <sub>H</sub> +x*10 <sub>H</sub>	<a href="#">17</a>
ID	Module Identification Register	0008 <sub>H</sub>	<a href="#">31</a>
LASRx	Last Acknowledged Service Request Register x, related to ICUx	0204 <sub>H</sub> +x*10 <sub>H</sub>	<a href="#">17</a>
LWSRx	Latest Winning Service Request Register x, related to ICUx	0200 <sub>H</sub> +x*10 <sub>H</sub>	<a href="#">16</a>
OIT	OTGM IRQ Trace	00A0 <sub>H</sub>	<a href="#">38</a>
OIXMS	OTGM IRQ MUX Missed IRQ Select	008C <sub>H</sub>	<a href="#">36</a>
OIXS0	OTGM IRQ MUX Select 0	0090 <sub>H</sub>	<a href="#">36</a>
OIXS1	OTGM IRQ MUX Select 1	0094 <sub>H</sub>	<a href="#">37</a>
OIXTS	OTGM IRQ MUX Trigger Set Select	0088 <sub>H</sub>	<a href="#">36</a>
OMISN	OTGM MCDS I/F Sensitivity Negedge	00A8 <sub>H</sub>	<a href="#">39</a>
OMISP	OTGM MCDS I/F Sensitivity Posedge	00A4 <sub>H</sub>	<a href="#">39</a>
OOBS	OTGM OTGB0/1 Status	0080 <sub>H</sub>	<a href="#">35</a>
OSSIC	OTGM SSI Control	0084 <sub>H</sub>	<a href="#">35</a>
SRBy	Service Request Broadcast Register x	0010 <sub>H</sub> +x*4	<a href="#">31</a>

**Table 560 Register Overview - SRC (sorted by Name)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Page Number</b>
SRCi	Service Request Control Register i	00000 <sub>H</sub> +i*4	<a href="#">4</a>

**Interrupt Router (IR)****16.13.1 System and ICU Control Registers****Module Identification Register**

Interrupt Router Module Identification Register.

ID	Module Identification Register	(0008 <sub>H</sub> )	Application Reset Value: 00B9 C0XX <sub>H</sub>
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			

	MOD_NUMBER	MOD_TYPE	MOD_REV
r		r	r

Field	Bits	Type	Description
<b>MOD_REV</b>	7:0	r	<b>Module Revision Number</b> This bit field defines the module revision number. The value of a module revision starts with 01 <sub>H</sub> (first revision).
<b>MOD_TYPE</b>	15:8	r	<b>Module Type</b> The bit field is set to C0 <sub>H</sub> which defines the module as a 32-bit module.
<b>MOD_NUMBE R</b>	31:16	r	<b>Module Number Value</b> This bit field defines a module identification number. The value for the Interrupt Router module is 009Bh.

**Service Request Broadcast Register x**

Interrupt Service Request Broadcast Registers can be used to trigger multiple Service Requests of a General Purpose Service Request Group in parallel.

**SRBx (x=0-5)**

Service Request Broadcast Register x	(0010 <sub>H</sub> +x*4)	Application Reset Value: 0000 0000 <sub>H</sub>
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16	0	
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	0	W W W W W W W W W W W W

Field	Bits	Type	Description
<b>TRIGi (i=0-7)</b>	i	w	<b>General Purpose Service Request Trigger i</b> This bit is always read as 0. 0 <sub>B</sub> No effect 1 <sub>B</sub> Trigger the General Purpose Service Request i in group x (GPSRx <sub>i</sub> )
<b>0</b>	31:8	r	<b>Reserved</b> Read as 0; should be written with 0.

**Interrupt Router (IR)****Access Enable covering all INT\_ECRx and all SRCy[15:0], Register 0**

Defines which TAG IDs from the range  $000000_B$  -  $011111_B$  are allowed to write to SRCy[15:0] (covering all implemented SRCs) and to the ECRx registers (all implemented Error Capture Registers). For the SRC protection pls. see also [Chapter 16.8.1](#).

**ACCEN\_CONFIG0**

**Access Enable covering all INT\_ECRx and all SRCy[15:0], Register 0(00F0<sub>H</sub>) Application Reset Value: FFFF  
FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

Field	Bits	Type	Description
ENx (x=0-31)	x	rw	<b>Access Enable for Master TAG ID x</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID x $0_B$ Write access will not be executed $1_B$ Write access will be executed

**Access Enable covering all INT\_ECRx and all SRCy[15:0], Register 1**

Defines which TAG IDs from the range  $100000_B$  -  $111111_B$  are allowed to write to SRCy[15:0] (covering all implemented SRCs) and to the ECRx registers (all implemented Error Capture Registers). For the SRC protection pls. see also [Chapter 16.8.1](#).

As these TAG IDs are not used in this product, this register is implemented as 'read only'. Write data will be silently ignored.

**ACCEN\_CONFIG1**

**Access Enable covering all INT\_ECRx and all SRCy[15:0], Register 1(00F4<sub>H</sub>) Application Reset Value: 0000  
0000<sub>H</sub>**

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
0																															

Field	Bits	Type	Description
0	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

**Access Enable covering SRBx, Register 0**

Defines which TAG IDs from the range  $000000_B$  -  $011111_B$  are allowed to write to INT\_SRBx (see also [Chapter 16.7.3](#)).

**Interrupt Router (IR)****ACCEN\_SRBy0 (x=0-5)****Access Enable covering SRBy, Register 0 (0100<sub>H</sub>+x\*8)****Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN31</b>	<b>EN30</b>	<b>EN29</b>	<b>EN28</b>	<b>EN27</b>	<b>EN26</b>	<b>EN25</b>	<b>EN24</b>	<b>EN23</b>	<b>EN22</b>	<b>EN21</b>	<b>EN20</b>	<b>EN19</b>	<b>EN18</b>	<b>EN17</b>	<b>EN16</b>
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>EN15</b>	<b>EN14</b>	<b>EN13</b>	<b>EN12</b>	<b>EN11</b>	<b>EN10</b>	<b>EN9</b>	<b>EN8</b>	<b>EN7</b>	<b>EN6</b>	<b>EN5</b>	<b>EN4</b>	<b>EN3</b>	<b>EN2</b>	<b>EN1</b>	<b>ENO</b>
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
<b>ENi (i=0-31)</b>	i	rw	<b>Access Enable for Master TAG ID i</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID i 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

**Access Enable covering SRBy, Register 1**

Defines which TAG IDs from the range 100000<sub>B</sub> - 111111<sub>B</sub> are allowed to write to INT\_SRBy (see also [Chapter 16.7.3](#)).

These TAG IDs are not used in this product. This register is therefore implemented as 'read only'. Write data will be silently ignored

**ACCEN\_SRBy1 (x=0-5)****Access Enable covering SRBy, Register 1 (0104<sub>H</sub>+x\*8)****Application Reset Value: 0000 0000<sub>H</sub>**

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

Field	Bits	Type	Description
<b>0</b>	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

**Access Enable covering all SRCx[31:16] mapped to ICUx, Register 0**

Defines for all SRNs that are mapped on TOSx (SRCy.TOS=x), which TAG IDs from the range 000000<sub>B</sub> - 011111<sub>B</sub> are allowed to write to the related SRC[31:15] (see also [Chapter 16.4.1.3](#)).

**Interrupt Router (IR)****ACCEN\_SRC\_TOSx0 (x=0-7)**

**Access Enable covering all SRCx[31:16] mapped to ICUx, Register 0(0180<sub>H</sub>+x\*8) Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

Field	Bits	Type	Description
ENi (i=0-31)	i	rw	<b>Access Enable for Master TAG ID i</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID i 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

**Access Enable covering all SRCx[31:16] mapped to ICUx, Register 1**

Defines for all SRNs that are mapped on TOSx (SRCy.TOS=x), which TAG IDs from the range 100000<sub>B</sub> - 111111<sub>B</sub> are allowed to write to the related SRC[31:15] (see also [Chapter 16.4.1.3](#)).

As these TAG IDs are not used in this product, this register is implemented as 'read only'. Write data will be silently ignored.

**ACCEN\_SRC\_TOSx1 (x=0-7)**

**Access Enable covering all SRCx[31:16] mapped to ICUx, Register 1(0184<sub>H</sub>+x\*8) Application Reset Value: 0000 0000<sub>H</sub>**

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
0																															

Field	Bits	Type	Description
0	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

**16.14 OTGM Registers**

All OTGM registers are cleared by Debug Reset and by each System Reset when OCDS is disabled. They are not touched by System Reset when OCDS is enabled.

Write access is 32 bit wide only and requires Supervisor Mode and OCDS enabled.

## Interrupt Router (IR)

#### **16.14.1 Status and Control**

## OTGM OTGB0/1 Status

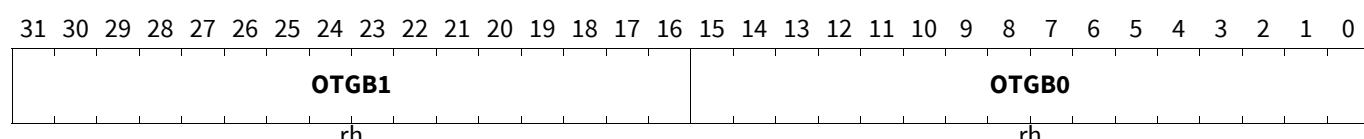
**Note:** OTGB0/1 value is sampled not captured. A capture register is available in OTGS.

OOBS

## OTGM OTGB0/1 Status

(0080<sub>H</sub>)

**Application Reset Value: 0000 0000.**



<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>OTGB0</b>	15:0	rh	<b>Status of OTGB0</b>
<b>OTGB1</b>	31:16	rh	<b>Status of OTGB1</b>

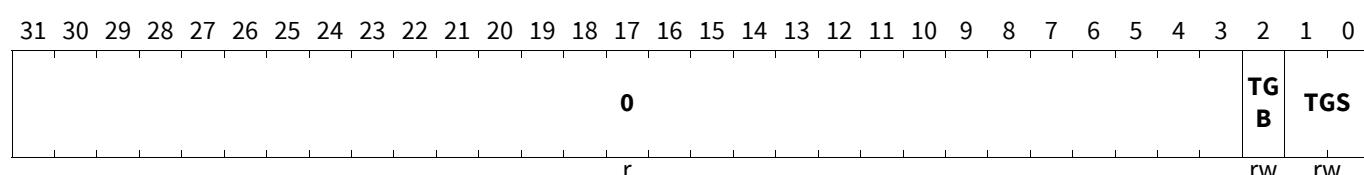
OTGM SSI Control

OSSIC

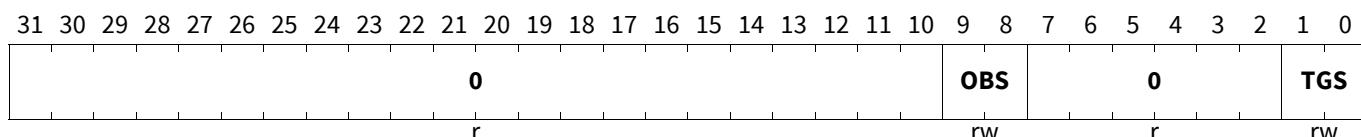
OTGM SSI Control

(0084<sub>II</sub>)

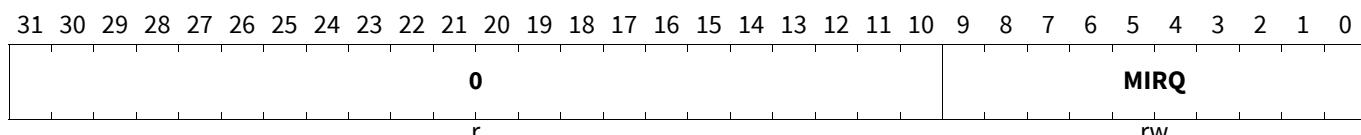
**Application Reset Value: 0000 0000..**



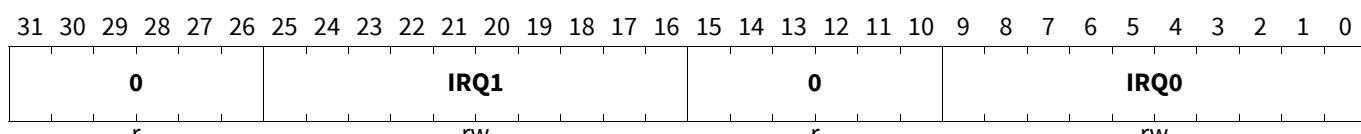
Field	Bits	Type	Description
<b>TGS</b>	1:0	rw	<b>Trigger Set for OTGB0/1</b> 00 <sub>B</sub> No Trigger Set output 01 <sub>B</sub> Trigger Set TS16_SSI <b>others</b> , reserved (no Trigger Set selected)
<b>TGB</b>	2	rw	<b>OTGB0/1 Bus Select</b> 0 <sub>B</sub> Trigger Set is output on OTGB0 1 <sub>B</sub> Trigger Set is output on OTGB1
<b>0</b>	31:3	r	<b>Reserved</b> Read as 0; must be written with 0.

**Interrupt Router (IR)****16.14.2 IRQ MUX Control****OTGM IRQ MUX Trigger Set Select****OIXTS****OTGM IRQ MUX Trigger Set Select****(0088<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>TGS</b>	1:0	rw	<b>Trigger Set Select for OTGB0/1 Overlay</b> $00_B$ No overlay $01_B$ Trigger Set TS8_IS $10_B$ Trigger Set TS8_SPA <b>others</b> , reserved (no Trigger Set selected)
<b>OBS</b>	9:8	rw	<b>Overlay Byte Select</b> $00_B$ OTGB0 [7:0] $01_B$ OTGB0 [15:8] $10_B$ OTGB1 [7:0] $11_B$ OTGB1 [15:8]
<b>0</b>	7:2, 31:10	r	<b>Reserved</b> Read as 0; must be written with 0.

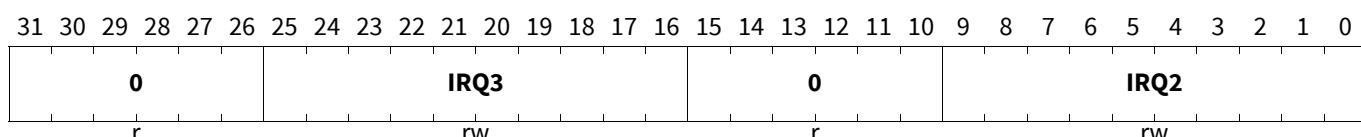
**OTGM IRQ MUX Missed IRQ Select****OIXMS****OTGM IRQ MUX Missed IRQ Select****(008C<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>MIRQ</b>	9:0	rw	<b>SRN Index for Missed Interrupt Trigger</b>
<b>0</b>	31:10	r	<b>Reserved</b> Read as 0; must be written with 0.

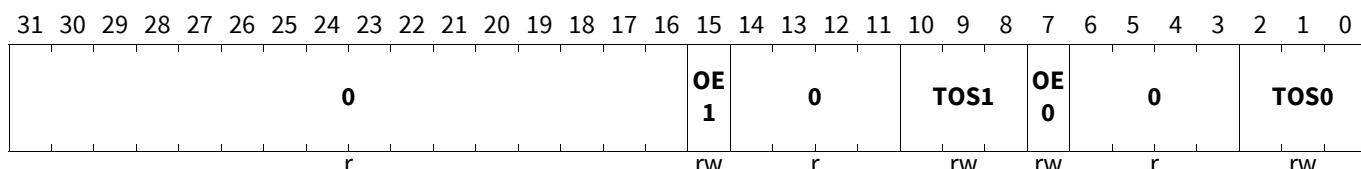
**OTGM IRQ MUX Select 0****OIXS0****OTGM IRQ MUX Select 0****(0090<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

**Interrupt Router (IR)**

Field	Bits	Type	Description
<b>IRQ0</b>	9:0	rw	<b>SRN Index for Interrupt Trigger 0</b>
<b>IRQ1</b>	25:16	rw	<b>SRN Index for Interrupt Trigger 1</b>
<b>0</b>	15:10, 31:26	r	<b>Reserved</b> Read as 0; must be written with 0.

**OTGM IRQ MUX Select 1****OIXS1****OTGM IRQ MUX Select 1**(0094<sub>H</sub>)**Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>IRQ2</b>	9:0	rw	<b>SRN Index for Interrupt Trigger 2</b>
<b>IRQ3</b>	25:16	rw	<b>SRN Index for Interrupt Trigger 3</b>
<b>0</b>	15:10, 31:26	r	<b>Reserved</b> Read as 0; must be written with 0.

**Interrupt Router (IR)****16.14.3 Interrupt System Trace****OTGM IRQ Trace****OIT****OTGM IRQ Trace****(00AO<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>TOS0</b>	2:0	rw	<b>Type of Service for Observation on OTGB0</b> Trigger Set TS16_SP Family concept encoding, compatible with SRC.TOS $000_B$ CPU0 service is observed $001_B$ DMA service is observed $010_B$ CPU1 service is observed $011_B$ CPU2 service is observed $100_B$ CPU3 service is observed $101_B$ CPU4 service is observed $110_B$ CPU5 service is observed <b>others</b> , Reserved (no action)
<b>OE0</b>	7	rw	<b>Output Enable for OTGB0</b> $0_B$ Disabled $1_B$ Enabled
<b>TOS1</b>	10:8	rw	<b>Type of Service for Observation on OTGB1</b> Trigger Set TS16_SP Family concept encoding, compatible with SRC.TOS $000_B$ CPU0 service is observed $001_B$ DMA service is observed $010_B$ CPU1 service is observed $011_B$ CPU2 service is observed $100_B$ CPU3 service is observed $101_B$ CPU4 service is observed $110_B$ CPU5 service is observed <b>others</b> , Reserved (no action)
<b>OE1</b>	15	rw	<b>Output Enable for OTGB1</b> $0_B$ Disabled $1_B$ Enabled
<b>0</b>	6:3, 14:11, 31:16	r	<b>Reserved</b> Read as 0; must be written with 0.

**Interrupt Router (IR)****16.14.4 MCDS Interface****OTGM MCDS I/F Sensitivity Posedge**

OMISP

<b>OTGM MCDS I/F Sensitivity Posedge</b>																<b>(00A4<sub>H</sub>)</b>																<b>Application Reset Value: 0000 0000<sub>H</sub></b>															
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																
OTGB1																OTGB0																rw				rw											

Field	Bits	Type	Description
OTGB0	15:0	rw	<b>Bitwise Posedge Sensitivity for OTGB0</b> If a bit is set an OTGB value will be written to MCDS on a rising edge of the associated OTGB0 bit.
OTGB1	31:16	rw	<b>Bitwise Posedge Sensitivity for OTGB1</b> If a bit is set an OTGB value will be written to MCDS on a rising edge of the associated OTGB1 bit.

**OTGM MCDS I/F Sensitivity Negedge**

OMISN

<b>OTGM MCDS I/F Sensitivity Negedge</b>																<b>(00A8<sub>H</sub>)</b>																<b>Application Reset Value: 0000 0000<sub>H</sub></b>															
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																
OTGB1																OTGB0																rw				rw											

Field	Bits	Type	Description
OTGB0	15:0	rw	<b>Bitwise Negedge Sensitivity for OTGB0</b> If a bit is set an OTGB value will be written to MCDS on a falling edge of the associated OTGB0 bit.
OTGB1	31:16	rw	<b>Bitwise Negedge Sensitivity for OTGB1</b> If a bit is set an OTGB value will be written to MCDS on a falling edge of the associated OTGB1 bit.

**16.15 Revision History****Table 561 Revision History**

Reference	Change to Previous Version	Comment
V1.2.6		
<a href="#">Page 19</a> , <a href="#">Page 25</a>	Number of GPSR per CPU: corrected two wrong comment regarding the number of GPSR per CPU (which is 8 in A2G).	
<a href="#">Page 39</a>	Removed old revision history lists.	
V1.2.7		

**Interrupt Router (IR)****Table 561 Revision History (cont'd)**

<b>Reference</b>	<b>Change to Previous Version</b>	<b>Comment</b>
	No functional changes	
<b>V1.2.8</b>		
	No functional changes	
<b>Page 3, Page 15</b>	Corrected 'Safety Monitor Unit' to 'Safety Management Unit'	
<b>V1.2.9</b>		
	No changes	
<b>V1.2.10</b>		
-	No functional changes.	

## Flexible CRC Engine (FCE)

### 17 Flexible CRC Engine (FCE)

The FCE provides a parallel implementation of Cyclic Redundancy Code (CRC) algorithms. The current FCE version for the AURIX™ TC3xx Platform implements the IEEE 802.3 ethernet CRC32, the Autosar CRC32P4, the CCITT CRC16 and the SAE J1850 CRC8 polynomials. The primary target of FCE is to be used as an hardware acceleration engine for software applications or operating systems services that use CRC signatures.

The FCE operates as a standard peripheral bus slave. The FCE supports multiple parallel channels of CRC context. Each of these channels can be configured to use any one of the CRC algorithms at a time. Thus, using these channels, multiple software tasks can use even the same CRC algorithm concurrently.

**Note:** The FCE register names described in “[Registers](#) on Page 15” are referenced in a product User’s Manual by the module name prefix “FCE<sub>x</sub>”, x: 0 - (Number of FCE instances - 1)

#### Input documents

- [D1] A painless guide to CRC Error Detection Algorithms, Ross N. Williams
- [D2] Autosar R3.1 Rev 0001, Specification of CRC Routines V3.0.2
- [D3] 32-Bit Cyclic Redundancy Codes for Internet Applications, Philip Koopman, International Conference on Dependable Systems and Networks (DSN), 2002

#### Related standards and norms

- [S1] IEEE 802.3 Ethernet 32-bits CRC

**Table 562 Abbreviations**

CRC	Cyclic Redundancy Checksum
FCE	Flexible CRC Engine
IR	Input Register
RES	Result
STS	Status
CFG	Configuration

#### 17.1 Feature List

The FCE provides the following features:

- The FCE implements the following CRC polynomials:
  - CRC kernel 0: IEEE 802.3 CRC32 ethernet polynomial: 0x04C11DB7<sup>1)</sup> -  
 $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$
  - CRC kernel 1: Autosar safety polynomial CRC32P4: 0xF4ACFB13 -  
 $x^{32}+x^{31}+x^{30}+x^{29}+x^{28}+x^{26}+x^{23}+x^{21}+x^{19}+x^{18}+x^{15}+x^{14}+x^{13}+x^{12}+x^{11}+x^9+x^8+x^4+x+1$
  - CRC kernel 2: CCITT CRC16 polynomial: 0x1021 -  $x^{16}+x^{12}+x^5+1$
  - CRC kernel 3: SAE J1850 CRC8 polynomial: 0x1D -  $x^8+x^4+x^3+x^2+1$
- Parallel CRC implementation
  - Data blocks to be computed by FCE shall be a multiple of the polynomial degree
  - Start address of Data blocks to be computed by FCE shall be aligned to the polynomial degree
- Parallel Channels of CRC context

1) The polynomial hexadecimal representation covers the coefficients (degree - 1) down to 0.

## Flexible CRC Engine (FCE)

- The FCE supports up to 8 parallel channels of CRC context.
- A channel can be configured to use any one of the implemented algorithms.
- Different channels can use the same or different CRC algorithms concurrently.
- Register Interface for CRC computation:
  - Input Register
  - CRC Register
  - Configuration Registers enabling to control the CRC operation and perform automatic checksum checks at the end of a message.
  - Extended register interface to control reliability of FCE execution in safety applications.
  - FCE supports endianness conversion.
- FCE can be reset independently by a module reset controlled by software. The reset affects all CRC channels.
- Error notification scheme via dedicated interrupt node for:
  - Transient error detection: error interrupt generation (maskable) with local status register (cleared by software)
  - Checksum failure: error interrupt generation (maskable) with local status register (cleared by software)
- FCE provides one interrupt line to the interrupt system.

## Flexible CRC Engine (FCE)

### 17.2 Overview

This chapter provides an overview of the features, applications and architecture of the FCE module.

#### 17.2.1 Application Mapping

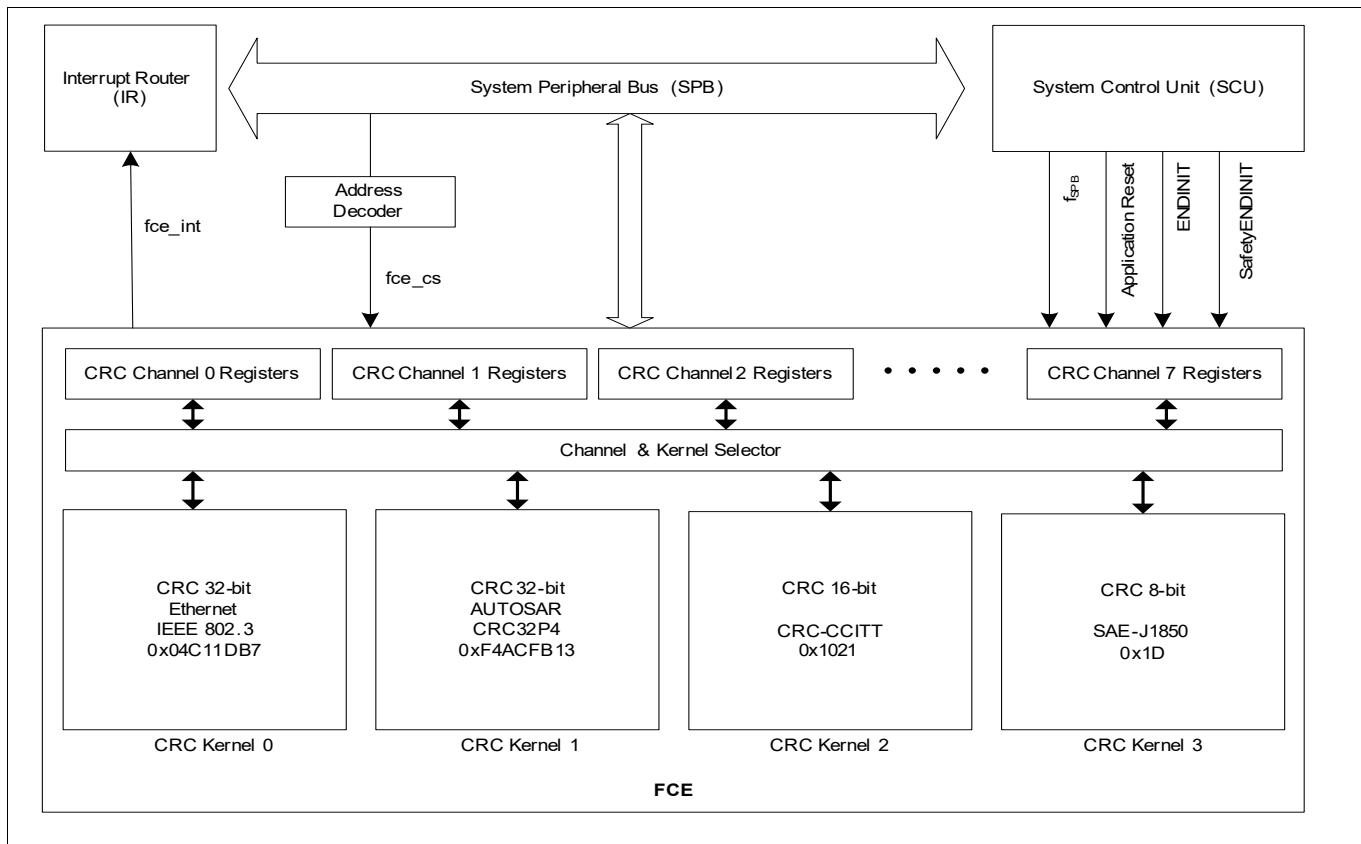
Among other applications, CRC algorithms are commonly used to calculate message signatures to:

- Check message integrity during transport over communication channels like internal busses or interfaces between micro-controllers
- Sign blocks of data residing in variable or invariable storage elements
- Compute signatures for program flow monitoring

#### 17.2.2 Block Diagram

The FCE is a standard peripheral slave module which is controlled over a set of memory mapped registers. The FCE is fully synchronous with the peripheral bus clock and runs with a 1:1 clock ratio.

The block diagram of the FCE in the AURIX™ TC3xx Platform is shown in the [Figure 180](#).



**Figure 180 FCE Block Diagram**

An FCE kernel is the instantiation of the CRC algorithm as a parallel matrix. Each FCE kernel implements a different CRC algorithm.

The FCE module contains 8 channels (Channel 0 - Channel 7). Each channel is provided with its own set of registers which allows the software to configure the channel, choose the algorithm kernel used, write the inputs and read the results and status from the CRC computation.

Each channel has an interrupt associated with it. The interrupt lines from all the channels are ORed together, and the FCE only presents a single interrupt node to the system. A status register associated with each channel

---

## Flexible CRC Engine (FCE)

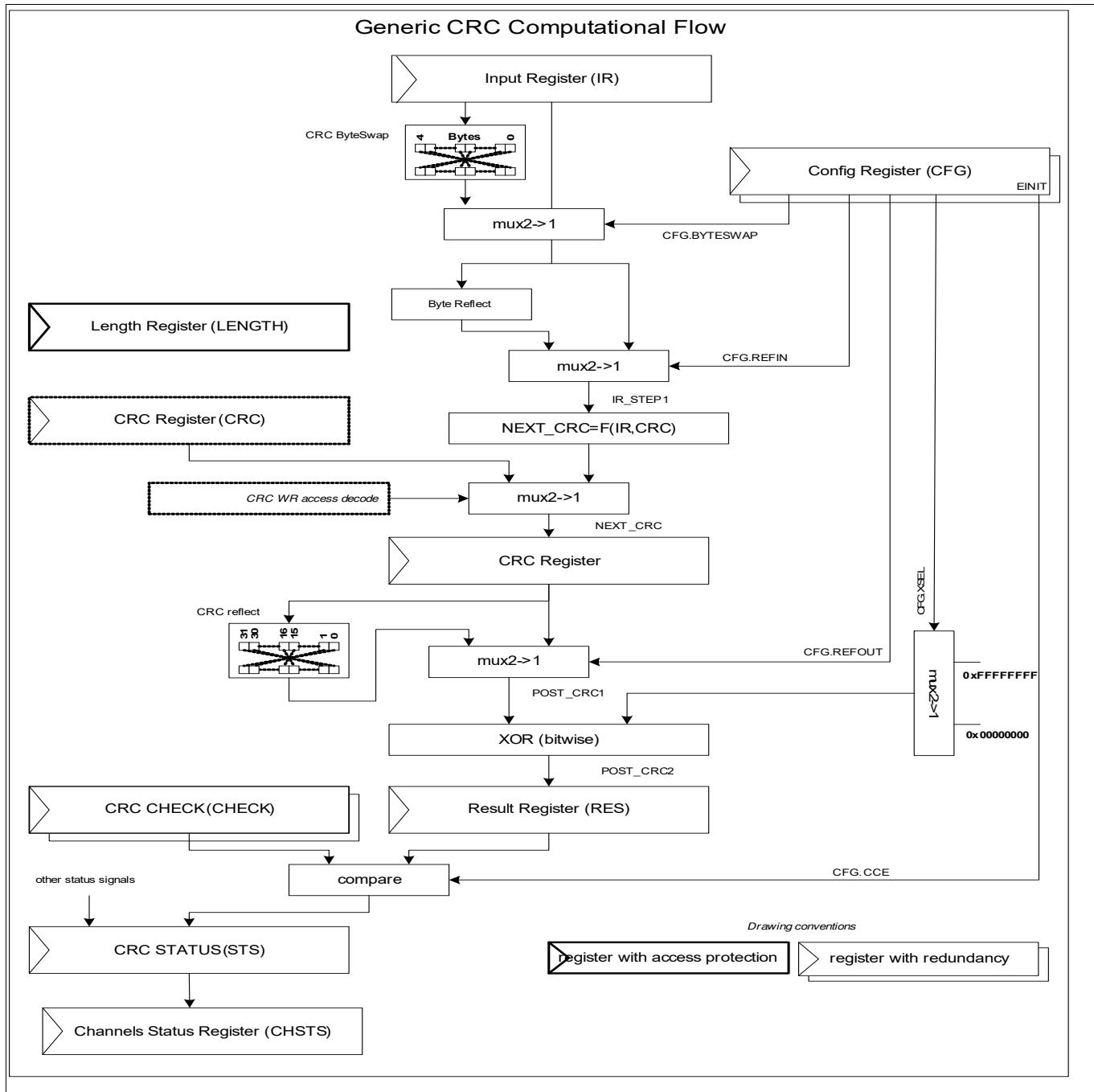
enables the software to identify which interrupt source is active. In addition, to aid fast interrupt handling in the software, a channels status register is provided, which basically ORs together the status from each channel. Thus the software can read this one register to get information regarding the pending status from each channel. Please refer to the **STS<sub>i</sub> (i=0-7)** register for a detailed description of the status and interrupt handling.

Each FCE channel presents the same hardware and software architecture for computing the CRC. The rest of this document will focus only on the description of the generic CRC computational flow, since each channel can be independently configured to use any CRC kernel.

## Flexible CRC Engine (FCE)

### 17.3 Functional Description

The generic CRC computational flow is presented in [Figure 181](#).



**Figure 181 CRC Computational flow**

A checksum algorithm based on CRC polynomial division is characterized by the following properties:

1. polynomial degree (e.g. 32, that represents the highest power of two of the polynomial)
2. polynomial (e.g. 0x04C11DB7: the 33rd bit is omitted because always equal to 1)
3. init value: the initial value of the CRC register
4. input data reflected: indicates if each byte of the input parallel data is reflected before being used to compute the CRC
5. result data reflected: indicates if the final CRC value is reflected or not.

## Flexible CRC Engine (FCE)

6. XOR value: indicates if a final XOR operation is done before returning the CRC result.
7. BYTESWAP: If set, the order of bytes in the input register (when 16- or 32-bit CRC is used) is swapped before the CRC conversion (For endianness conversion).

All the properties are fixed once a polynomial has been chosen. However the FCE provides the capability to control the two reflection steps and the final XOR as depicted in [Figure 181](#) through the CFG register. The reset values are compatible with the implemented algorithm. The final xor control enables to select either 0xFFFFFFFF or 0x00000000 to be xored with the POST\_CRC1 (see [Figure 181](#)) value. These two values are those used by the most common CRC polynomials.

As shown in the figure, when the software writes a value to the IR register, the CRC register is updated in the next clock cycle. The RES register is updated after the output operations (REFOUT, XSEL) still one more cycle later. (ie, there are 2 cycles from a write to the IR until the result gets updated in the RES register).

It shall be ensured by the programmer that the CFG register is configured before starting the CRC computation, and is not changed until the final result is obtained.

Note that the CRC computation of the FCE is equivalent to a bitwise shift CRC computation considering that the MSB bit is shifted in first.

**Note:** *The reflection steps and final XOR do not modify the properties of the CRC algorithm in terms of error detection, only the CRC final signature is affected.*

The next two figures provides an overview of the control and status features of a CRC channel.

## Flexible CRC Engine (FCE)

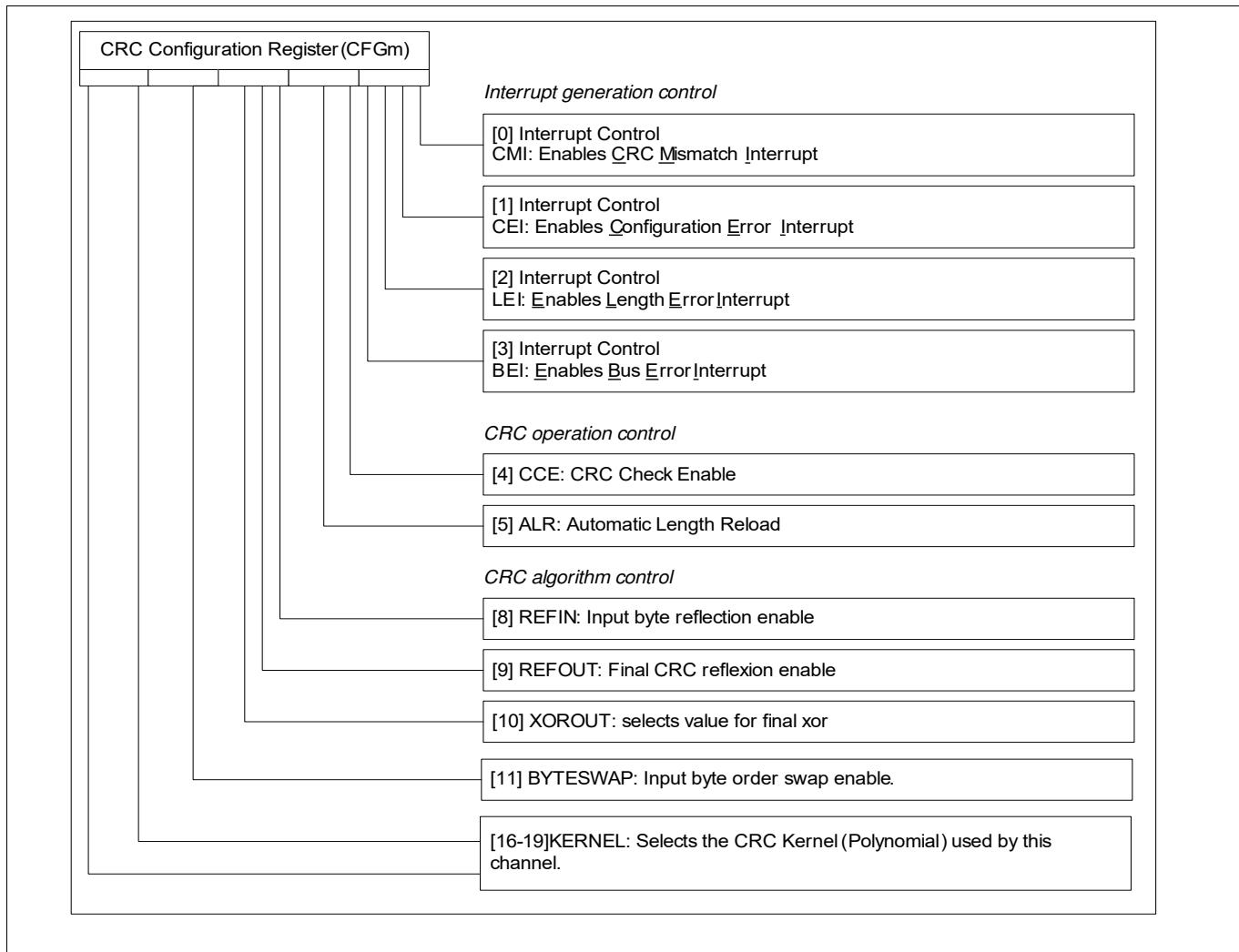


Figure 182 CRC Channel configuration register

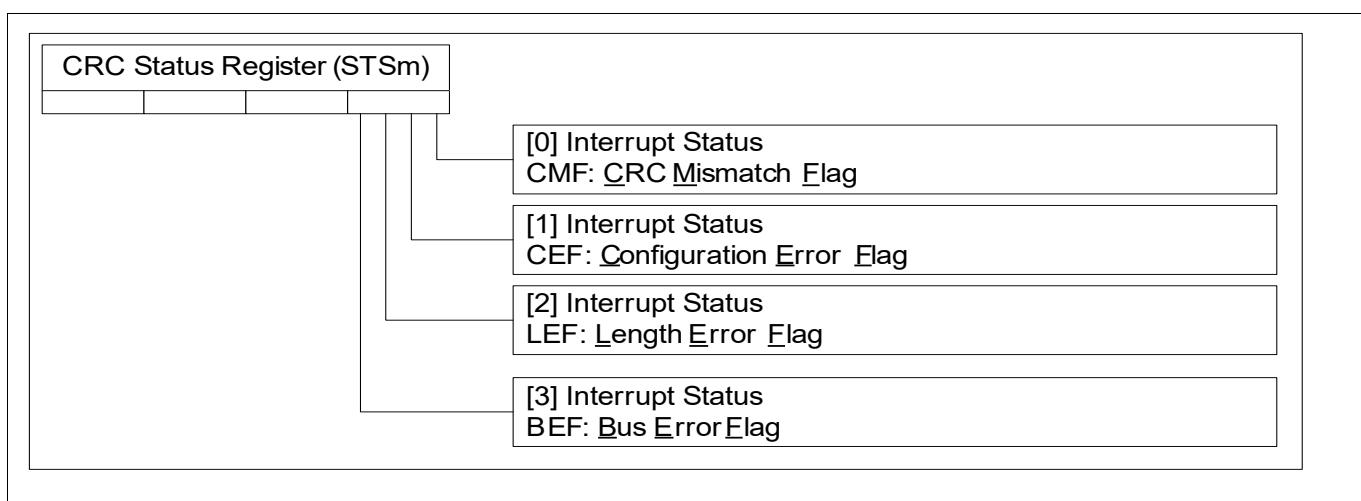


Figure 183 CRC kernel status register

## Flexible CRC Engine (FCE)

### 17.3.1 Initialization

The FCE is enabled by writing 0x0 to the CLC register. Software must first ensure that the CRC channel is properly configured, especially the initial CRC register value written via the CRC register, the CRC kernel, input and result reflection as well as the final xored value via the CFG register. The following source code is an example of initialization for the basic operation of the FCE channel 0:

```
//enable FCE
FCE_CLC.U = 0x0;
//Use Kernel-1, final result to be xored with 0xFFFFFFFF, no reflection
FCE_CFG0.U = 0x10400;
//set CRC initial value (seed)
FCE_CRC0.U = 0xFFFFFFFF;
```

### 17.3.2 Basic Operation

The software must first ensure that the CRC channel is properly configured; especially 1)The configuration, including the CRC kernel used in the CFG register; 2) Length of the data stream in the LENGTH register; and 3) The initial CRC value written via the CRC register. Then, it writes as many times as necessary into the IR register according to the length of the message. The resulting signature is stored in the result register, RES, which can be read by the software.

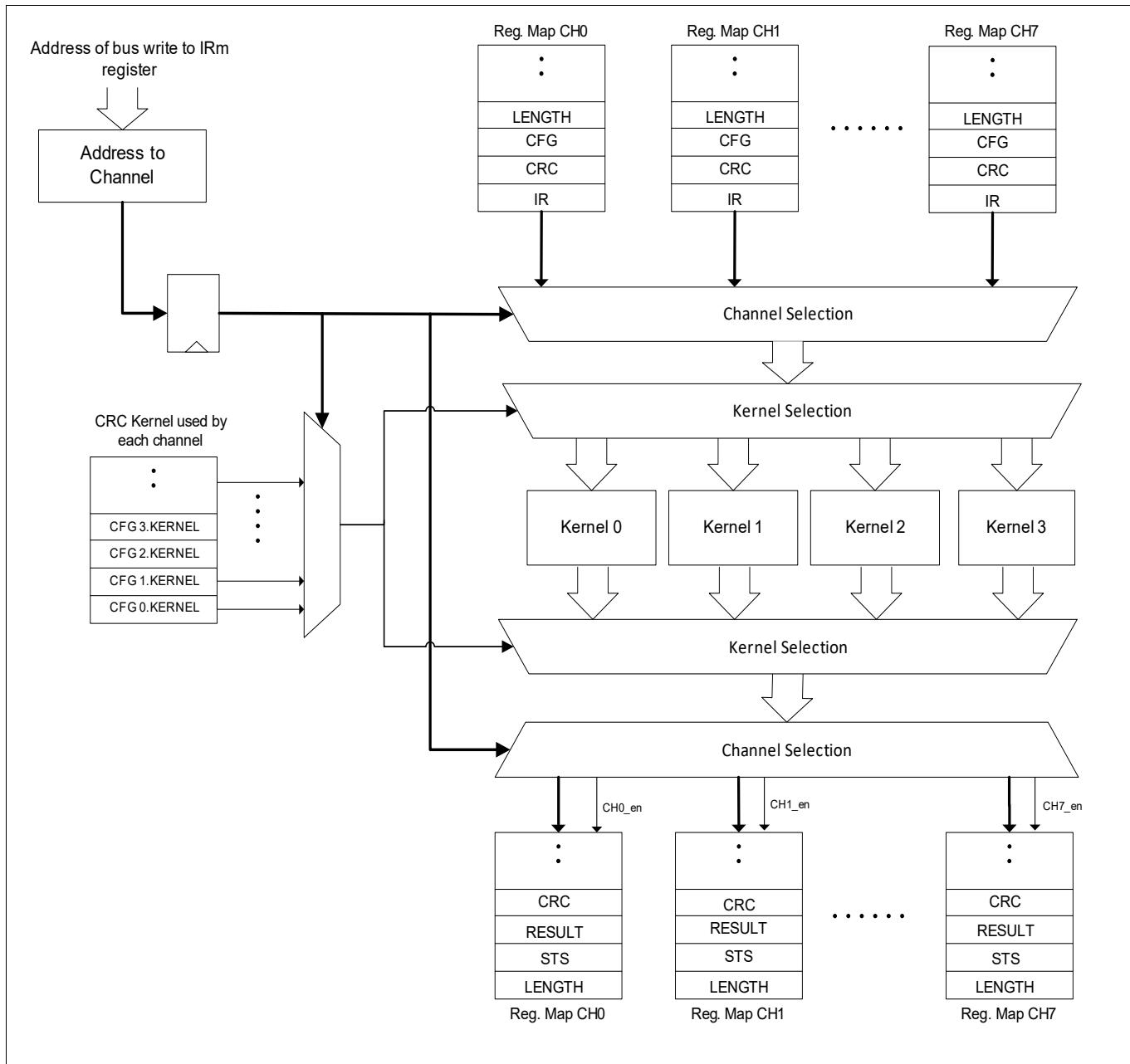
Depending on the CRC kernel that is configured to be used by the channel, the following rules apply:

- When using a CRC kernel of degree  $<N>$  only the bits N-1 down to 0 are used. The upper bits are ignored on write. When reading from a CRC kernel register the non-used upper bits are set to 0.

Each of the 8 channels can be configured to use any of the implemented CRC algorithm kernels. Multiple channels may be configured to use the same kernel, thereby enabling concurrent access of the same CRC algorithm by software.

When software writes to an IR register, based on the bus address of the IR register, the registers of the corresponding channel are selected to be used in the CRC computation (shown in [Figure 181](#)). The CRC kernel used by this channel is selected based on the CFG.KERNEL bits of the selected channel (see [CFG<sub>i</sub> \(\*i\*=0-7\)](#)). The logical representation of the channel and kernel selection based on the bus address of the IR register is shown in [Figure 184](#).

## Flexible CRC Engine (FCE)



**Figure 184 FCE channel and kernel selection**

### 17.3.3 Automatic Signature Check

The automatic signature check compares the signature at the end of a message with the expected signature configured in the CHECK register. In case of a mismatch, an event is generated (see [Section 17.3.7](#)). This feature is enabled by the CFG.CCE bit field (see [CFG<sub>i</sub> \(i=0-7\)](#) register).

If the software wishes to use this feature, the LENGTH register and CHECK registers of the channel must be configured with respectively the length as number of words of the message and the expected signature (CHECK). The word length is defined by the degree of the polynomial used. The CHECK value takes into account the final CRC reflection and XOR operation.

When the CFG.CCE bit field is set, every time the IR register is written, the LENGTH register is decremented by one until it reaches zero. The hardware monitors the transition of the LENGTH register from 1 to 0 to detect the end of the message and proceed with the comparison of the result register RES (see [Figure 181](#)) value with the CHECK register value. If the automatic length reload feature is enabled by the CFG.ALR bit field (see [CFG<sub>i</sub> \(i=0-7\)](#)), the

## Flexible CRC Engine (FCE)

LENGTH register is reinitialized with the previously configured value. This feature is especially suited when the FCE is used in combination with a DMA engine.

In the case the automatic length reload feature is not enabled, if LENGTH is already at zero but software still writes to IR (by mistake) every bit of the LENGTH is set to 1 this value is held until software initializes it again for the processing of a new message. In such case the STS.LEF (Length Error Flag) is set and an interrupt generated if the CFG.LEI (Length Error Interrupt) is set, and the corresponding written value is completely ignored, and no CRC calculation update done. In such an error case, further decrement and reload of the length as well as CRC check is disabled. Software shall clear the LEF error flag for further correct operation.

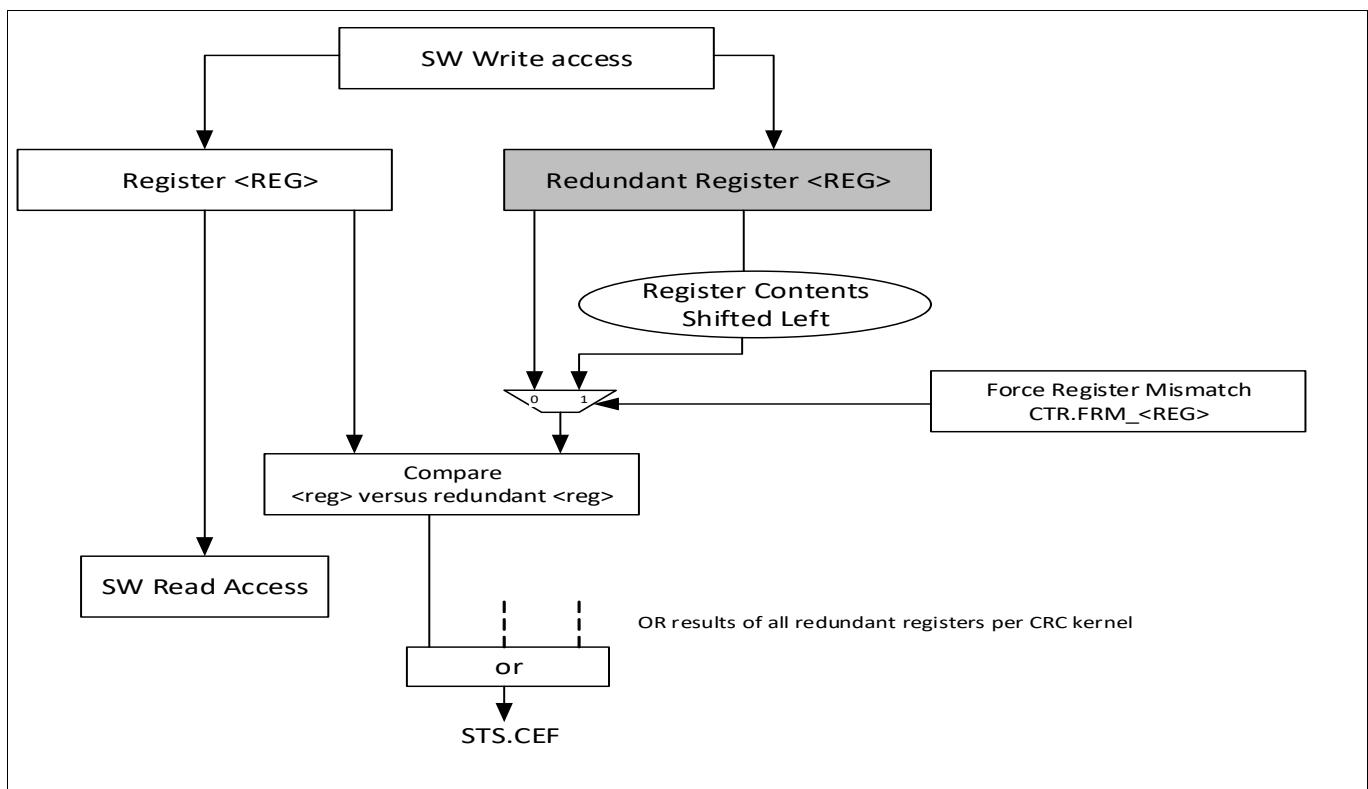
Usually, the CRC signature of a message M0 is computed and appended to M0 to form the message M1 which is transmitted. One interesting property of CRCs is that the CRC signature of M1 shall be zero. This property is particularly useful when automatically checking the signature of data blocks of fixed length with the automatic length reload enabled. LENGTH should be loaded with the length of M1 and CHECK with 0.

## Flexible CRC Engine (FCE)

### 17.3.4 Register protection and monitoring methods

Register Monitoring: applied to CFG and CHECK registers

Because CFG and CHECK registers are critical to the CRC operation, some mechanisms to detect and log transient errors are provided. Early detection of transient failures enables to improve the failure detection time and assess the severity of the failure. The monitoring mechanisms are implemented using two redundant instances as presented in [Figure 185](#).



**Figure 185 Register monitoring scheme**

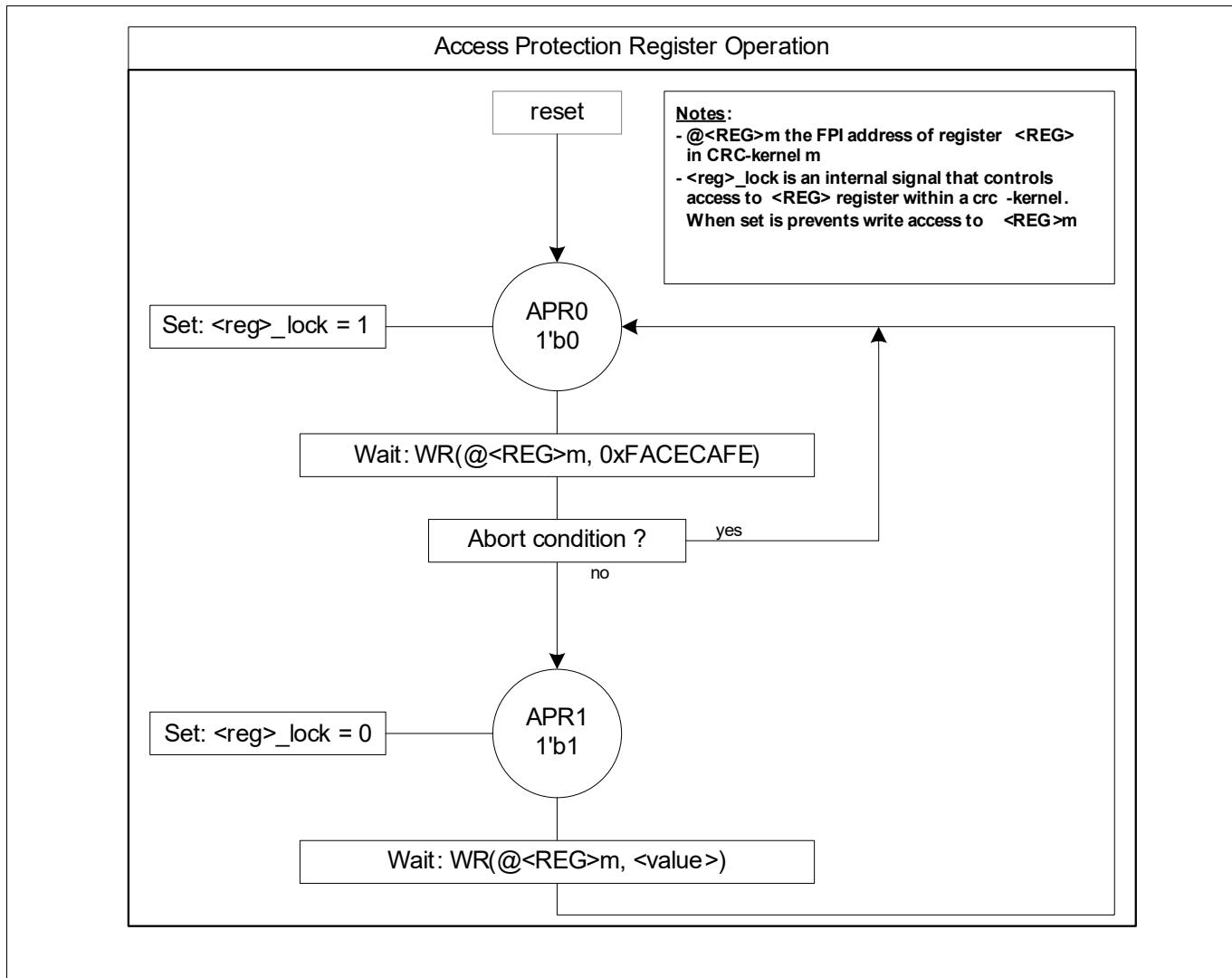
Let  $\langle \text{REG} \rangle$  designate either CFG or CHECK registers. When a write to  $\langle \text{REG} \rangle$  takes place the redundant register is also updated. Redundant registers are not visible to software. Bits of  $\langle \text{REG} \rangle$  reserved have no storage and are not used for redundancy. A compare logic continuously compares the two stored values and provides a signal that indicates if the compare is successful or not. The result of all compare blocks are ored together to provide a single flag information. If a mismatch is detected the STS.CEF (Configuration Error Flag) bit is set. For run-time validation of the compare logic a Force Register Mismatch bit field (CTR.FRM\_<REG>) is provided. When set to 1 by software the contents of the redundant register is shifted left by one bit position (redundant bit 0 position is always replaced by a logical 0 value) and is given to the compare logic instead of the redundant register value. This enables to check the compare logic is functional. Using a walking bit pattern, the software can completely check the full operation of the compare logic. Software needs to clear the CTR.FRM\_<REG> bit to '0' to be able to trigger again a new comparison error interrupt.

Register Access Protection: applies to LENGTH and CHECK registers

In order to reduce the probability of a mis-configuration of the CHECK and LENGTH registers (in the case the automatic check is used), the write access to the CHECK and LENGTH registers must follow a procedure:

This procedure is depicted in [Figure 186](#):

## Flexible CRC Engine (FCE)



**Figure 186 Access control to CHECK register**

Let <REG> designate CHECK or LENGTH registers. Before being able to configure a new <value> value into the <REG> register of a CRC kernel, software must first write the 0xFACECAFE value to the <REG> address. The 0xFACECAFE is not written into the <REG> register. The next write access will proceed as a normal bus write access. The write accesses shall use full 32-bit access only. This procedure will then be repeated every time software wants to configure a new <REG> value. If software reads the CHECK register just after writing 0xFACECAFE it returns the current <REG> contents and not 0xFACECAFE. A read access to <REG> has no effect on the protection mechanism.

The following C-code shows write accesses to the CHECK and LENGTH registers following this procedure:

```

//set CHECK register
FCE_CHECK0.U = 0xFACECAFE;
FCE_CHECK0.U = 0;
//set LENGTH register
FCE_LENGTH0.U = 0xFACECAFE;
FCE_LENGTH0.U = 256;

```

---

## Flexible CRC Engine (FCE)

### 17.3.5 Power, Reset and Clock

The FCE is inside the core power domain, therefore no special considerations about power up or power down sequences need to be taken. For an explanation about the different power domains, please address the SCU (System Control Unit) chapter.

A power down mode can be achieved by disabling the module using the **CLC** register.

The FCE module has one reset source. This reset source is handled at system level and it can be generated independently via a system control register (address SCU chapter for full description).

After reset, the complete IP is set to default configuration. The default configuration for each register field is addressed on [Section 17.4](#).

## Flexible CRC Engine (FCE)

### 17.3.6 Properties of CRC code

#### Hamming Distance

The Hamming distance defines the error detection capability of a CRC polynomial. A cyclic code with a Hamming Distance of D can detect all D-1 bit errors. **Table 563** shows the dependency of the Hamming Distance with the length of the message.

**Table 563 Hamming Distance as a function of message length (bits)<sup>1)</sup>**

Hamming Distance	IEEE-802.3 CRC32	CCITT CRC16	J1850 CRC8
15	8 - 10	Information not available	Information not available
14	8 - 10		
13	8 - 10		
12	11 - 12		
11	13 - 21		
10	22 - 34		
9	35 - 57		
8	58 - 91		
7	92 - 171		
6	172 - 268		
5	269 - 2974		
4	2975 - 91606		
3	91607 - 131072		

1) Data from technical paper "32-Bit Cyclic Redundancy Codes for Internet Applications" by Philip Koopman, Carnegie Mellon University, 2002

### 17.3.7 Service Request Generation

Each FCE CRC channel provides one internal interrupt source. The interrupt lines from each CRC channel are ORed together to be sent to the interrupt system. The system interrupt is an active high pulse with the duration of one cycle (of the peripheral clock). The FCE interrupt handler can use the status information located within the STS status register of each CRC channel.

Each CRC channel provides the following interrupt sources:

- CRC Mismatch Interrupt controlled by CFG.CMI bit field and observable via the status bit field STS.CMF (CRC Mismatch Flag).
- Configuration Error Interrupt controlled by CFG.CEI bit field and observable via the status bit field STS.CEF (Configuration Error Flag).
- Length Error Interrupt controlled by CFG.LEI bit field and observable via the status bit field STS.LEF (Length Error Flag).
- Bus Error Interrupt controlled by CFG.BEI bit field and observable via the status bit field STS.BEF (Bus Error Flag). This error is triggered by a write to the IR register with write width less than the selected kernel's polynomial width.

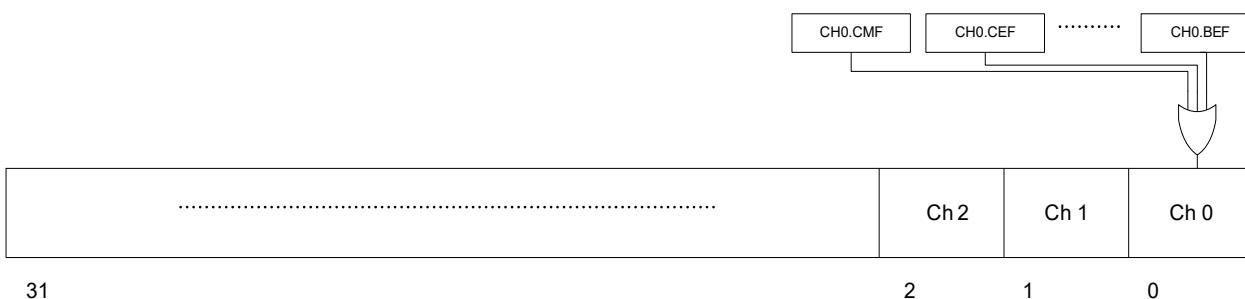
Interrupt generation rules

## Flexible CRC Engine (FCE)

- A status flag shall be cleared by software by writing a 0 to the corresponding bit position in the STS register.
- If an status flag is set and a new hardware condition occurs, no new interrupt is generated by the kernel: a set STS.<FLAG> bit field masks the generation of a new interrupt from the FCE module. If a SW access to clear the interrupt status bit takes place and in the same cycle the hardware wants to set the bit, the hardware condition wins the arbitration.

To aid faster interrupt handling in the software, the status flags from all channels each channel are ORed together and provided as individual bits in a Channels Status Register (see [Figure 187](#)). The software can read this single register to learn the channel status, rather than reading the registers of each channel separately.

Since there is only a single interrupt from the FCE, the software has to clear the status flags from all the channels to forward a further FCE interrupt.



**Figure 187 Channels Status Register**

## 17.4 Registers

[Table 564](#) show all registers associated with the FCE module. All FCE channel register names are described in this section. They should get the prefix “FCE\_” when used in the context of a product specification.

The registers are numbered by one index to indicate the related FCE CRC Channel ( $m = 0\text{-}7$ ).

[Figure 188](#) shows the FCE module register map.

## Flexible CRC Engine (FCE)

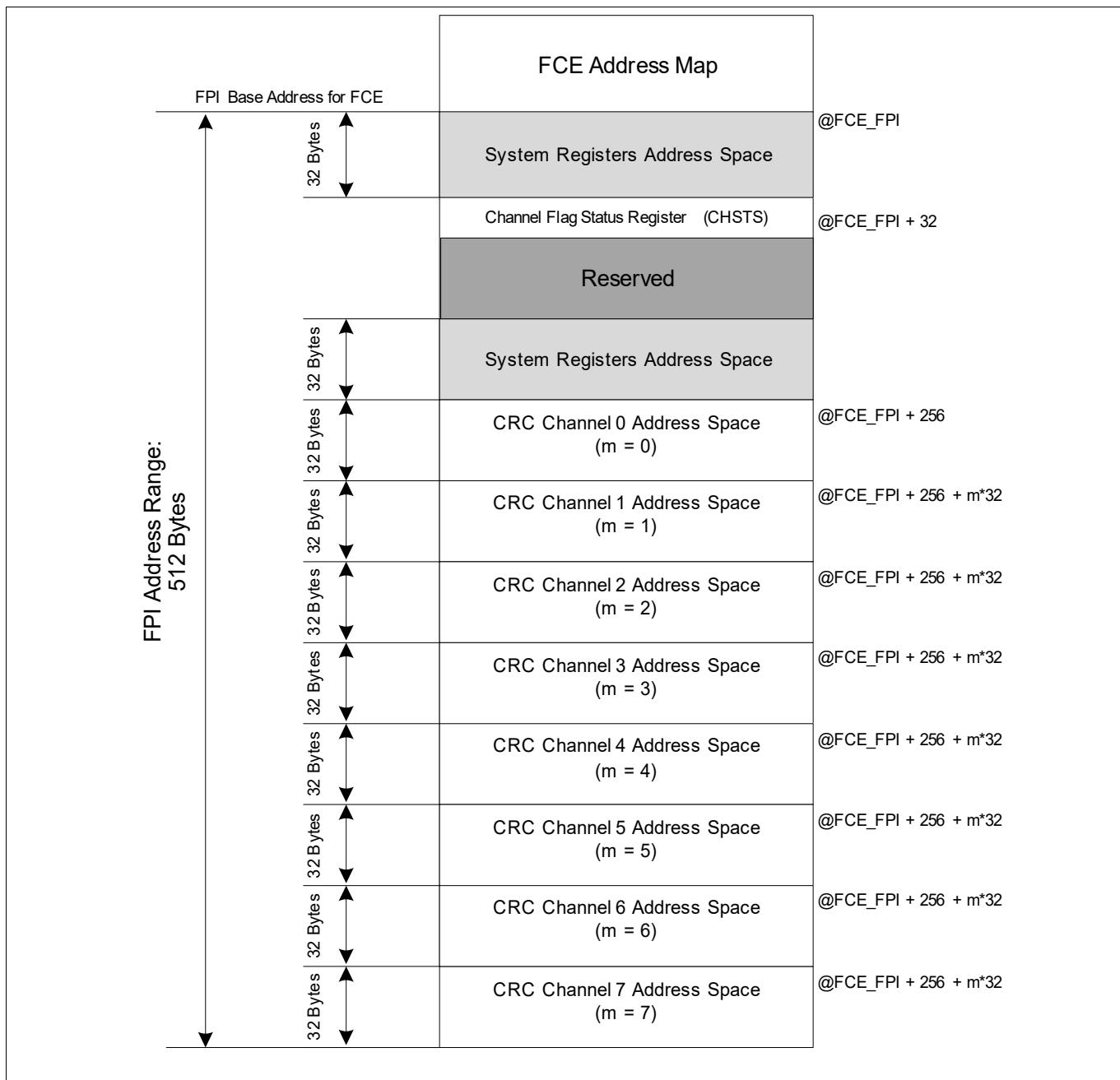


Figure 188 FCE Registers Address Map

Table 564 Register Overview - FCE (ascending Offset Address)

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
CLC	Clock Control Register	000 <sub>H</sub>	U,SV	E,SV,P	Application Reset	<a href="#">19</a>
ID	Module Identification Register	008 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">19</a>
CHSTS	Channels Status Register	020 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">23</a>

## Flexible CRC Engine (FCE)

**Table 564 Register Overview - FCE (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
KRSTCLR	Kernel Reset Status Clear Register	0EC <sub>H</sub>	U,SV	SV,E,P	Application Reset	<b>20</b>
KRST1	Kernel Reset Register 1	0F0 <sub>H</sub>	U,SV	SV,E,P	Application Reset	<b>20</b>
KRST0	Kernel Reset Register 0	0F4 <sub>H</sub>	U,SV	SV,E,P	Application Reset	<b>21</b>
ACCEN1	Access Enable Register 1	0F8 <sub>H</sub>	U,SV	SV,SE	Application Reset	<b>22</b>
ACCENO	Access Enable Register 0	0FC <sub>H</sub>	U,SV	SV,SE	Application Reset	<b>22</b>
IRi	Input Register i	100 <sub>H</sub> +i*2 0 <sub>H</sub>	U,SV	P,U,SV	Application Reset	<b>25</b>
RESi	CRC Result Register i	104 <sub>H</sub> +i*2 0 <sub>H</sub>	U,SV	BE	Application Reset	<b>25</b>
CFGi	CRC Configuration Register i	108 <sub>H</sub> +i*2 0 <sub>H</sub>	U,SV	P,E,SV	Application Reset	<b>26</b>
STSi	CRC Status Register i	10C <sub>H</sub> +i*2 0 <sub>H</sub>	U,SV	P,U,SV	Application Reset	<b>27</b>
LENGTHi	CRC Length Register i	110 <sub>H</sub> +i*2 0 <sub>H</sub>	U,SV	P,U,SV	Application Reset	<b>28</b>
CHECKi	CRC Check Register i	114 <sub>H</sub> +i*2 0 <sub>H</sub>	U,SV	P,U,SV	Application Reset	<b>29</b>
CRCi	CRC Register i	118 <sub>H</sub> +i*2 0 <sub>H</sub>	U,SV	P,U,SV	Application Reset	<b>29</b>
CTRi	CRC Test Register i	11C <sub>H</sub> +i*2 0 <sub>H</sub>	U,SV	P,U,SV	Application Reset	<b>30</b>

### Access Mode Rules

The **Table 564** uses the standard access mode conventions.

### Disabling the FCE

The FCE module can be disabled using the **CLC** register.

When the disable state is requested all pending transactions running on the bus slave interface must be completed before the disabled state is entered. The CLC Register Module Disable Bit Status CLC.DISS indicates whether the module is currently disabled (DISS == 1). Any attempt to write any of the BPI writable registers with the exception of the CLC Register will generate a bus error. A read operation of BPI registers is allowed and does not generate a bus error.

### Resetting the FCE

The FCE module can be reset using the **KRST0**, **KRST1**, and **KRSTCLR** registers. This action affects all the CRC kernels and channels.

## Flexible CRC Engine (FCE)

To reset the FCE it is necessary to set the RST bits by writing with '1' in both Reset Registers **KRST0**, **KRST1**. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Reset Register 0 includes a reset status bit that is set to '1' by the BPI in the same clock cycle the RST bit is re-set by the BPI. This bit can be used to detect that a reset was processed. The bit can be re-set to '0' by writing to KRSTCLR.CLR with '1'.

Additionally, when a channel is reprogrammed to use a different kernel, then software should take care to reset or reprogram the channel registers (esp. IR, RES, CRC and CHECK). Otherwise masked bits (eg. for a Lower width polynomial Kernel) may become visible and used.

## Flexible CRC Engine (FCE)

### 17.4.1 System Registers description

This section describes the registers related to the product system architecture.

#### Clock Control Register

The Clock Control Register allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

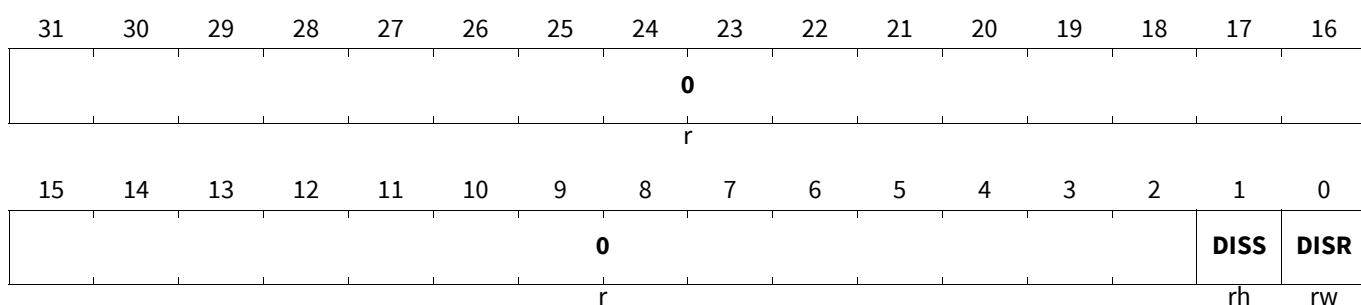
**Note:** *The features related to the pre-defined FDIS, EDIS and RMC fields are not supported by the FCE. Therefore the corresponding fields have been removed from the CLC register.*

#### CLC

##### Clock Control Register

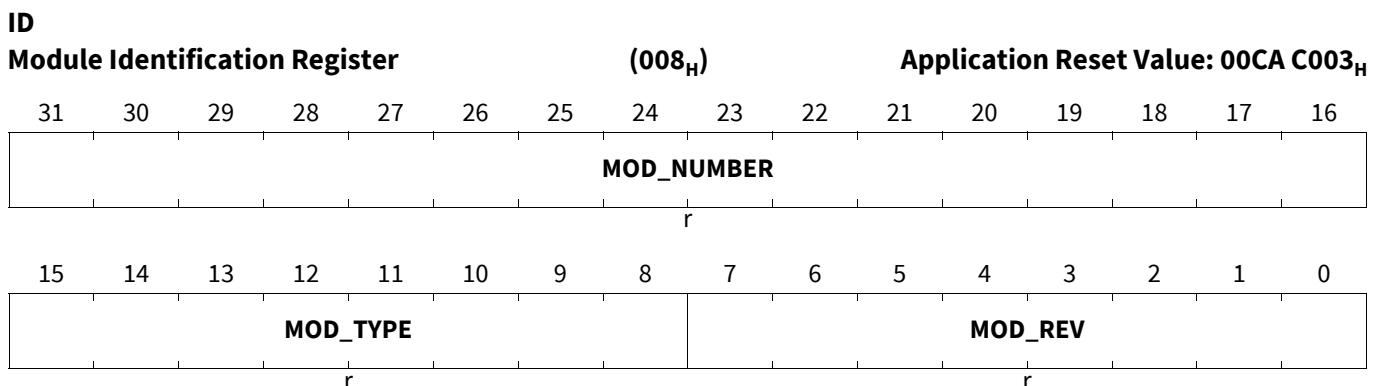
(000<sub>H</sub>)

Application Reset Value: 0000 0003<sub>H</sub>



Field	Bits	Type	Description
DISR	0	rw	<b>Module Disable Request Bit</b> Used for enable/disable control of the module.
DISS	1	rh	<b>Module Disable Status Bit</b> Bit indicates the current status of the module.
0	31:2	r	<b>Reserved</b> Read as 0; should be written with 0.

#### Module Identification Register



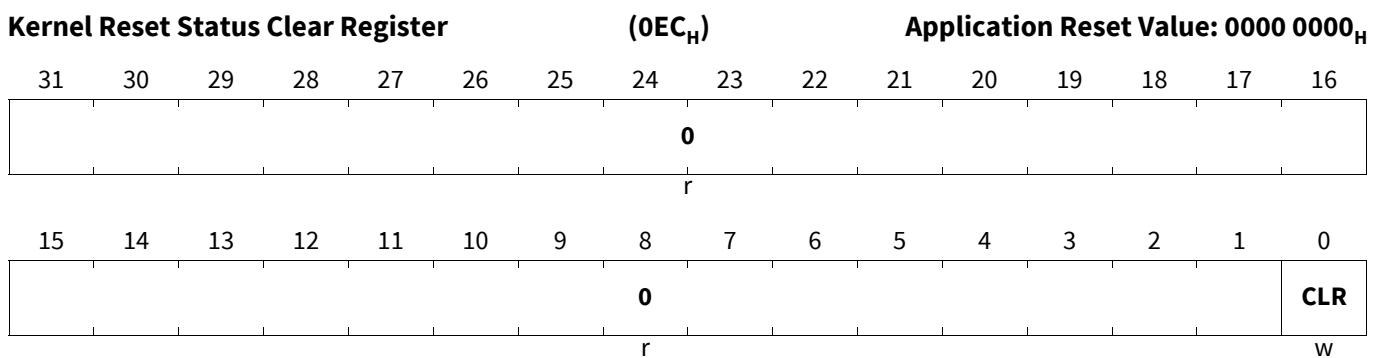
## Flexible CRC Engine (FCE)

Field	Bits	Type	Description
<b>MOD_REV</b>	7:0	r	<b>Module Revision Number</b> This bit field defines the module revision number. The value of a module revision starts with 01H (first revision). The current revision number is 03H.
<b>MOD_TYPE</b>	15:8	r	<b>Module Type</b> The bit field is set to C0H which defines the module as a 32-bit module.
<b>MOD_NUMBE R</b>	31:16	r	<b>Module Number Value</b> This bit field defines a module identification number. The value for the FCE module is 00CA <sub>H</sub> .

### Kernel Reset Status Clear Register

Refer to “[Resetting the FCE](#)” on Page 17 for the usage of the register.

#### KRSTCLR



Field	Bits	Type	Description
<b>CLR</b>	0	w	<b>Kernel Reset Status Clear</b> Read always as 0. 0 <sub>B</sub> No action 1 <sub>B</sub> Clear Kernel Reset Status KRST0.RSTSTAT
<b>0</b>	31:1	r	<b>Reserved</b> Read as 0; should be written with 0.

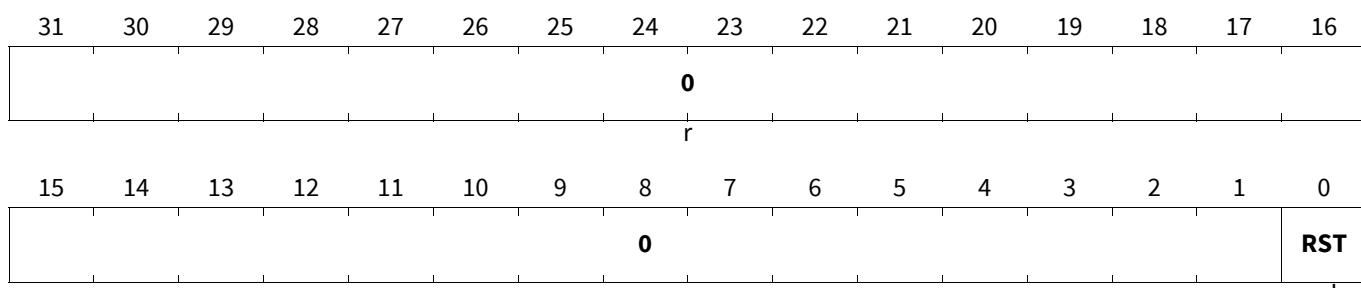
### Kernel Reset Register 1

Refer to “[Resetting the FCE](#)” on Page 17 for the usage of the register.

## Flexible CRC Engine (FCE)

### KRST1

#### Kernel Reset Register 1

(0F0<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

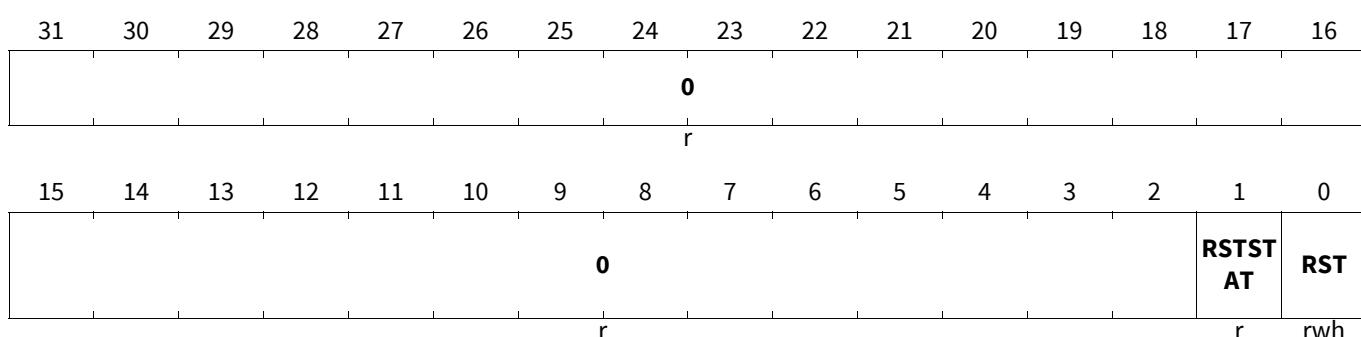
Field	Bits	Type	Description
RST	0	rwh	<b>Kernel Reset</b> This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed. 0 <sub>B</sub> No kernel reset was requested 1 <sub>B</sub> A kernel reset was requested
0	31:1	r	<b>Reserved</b> Read as 0; should be written with 0.

### Kernel Reset Register 0

Refer to “Resetting the FCE” on Page 17 for the usage of the register.

### KRST0

#### Kernel Reset Register 0

(0F4<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
RST	0	rwh	<b>Kernel Reset</b> This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed. 0 <sub>B</sub> No kernel reset was requested 1 <sub>B</sub> A kernel reset was requested

## Flexible CRC Engine (FCE)

Field	Bits	Type	Description
RSTSTAT	1	r	<p><b>Kernel Reset Status</b></p> <p>This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.</p> <p>0<sub>B</sub> No kernel reset was executed 1<sub>B</sub> Kernel reset was executed</p>
0	31:2	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

### Access Enable Register 1

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B.

#### ACCEN1

(0F8 <sub>H</sub> )																Application Reset Value: 0000 0000 <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	0															

Field	Bits	Type	Description
0	31:0	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

### Access Enable Register 0

The Access Enable Register 0 controls write access for transactions with the on chip bus master TAG ID 000000B to 011111B.

#### ACCENO

(0FC <sub>H</sub> )																Application Reset Value: FFFF FFFF <sub>H</sub>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	EN31 EN30 EN29 EN28 EN27 EN26 EN25 EN24 EN23 EN22 EN21 EN20 EN19 EN18 EN17 EN16															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	EN15 EN14 EN13 EN12 EN11 EN10 EN9 EN8 EN7 EN6 EN5 EN4 EN3 EN2 EN1 EN0															

## Flexible CRC Engine (FCE)

Field	Bits	Type	Description
<b>ENn (n=0-31)</b>	n	rw	<b>Access Enable for Master TAG ID n</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

### 17.4.2 FCE Common Registers

#### Channels Status Register

##### CHSTS

##### Channels Status Register

(020<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
r								rh							

Field	Bits	Type	Description
<b>CH0</b>	0	rh	<b>Channel0 Status</b> This bit is the result of an OR operation of the various status bits of channel 0 (see STS register).
<b>CH1</b>	1	rh	<b>Channel1 Status</b> This bit is the result of an OR operation of the various status bits of channel 1 (see STS register).
<b>CH2</b>	2	rh	<b>Channel2 Status</b> This bit is the result of an OR operation of the various status bits of channel 2 (see STS register).
<b>CH3</b>	3	rh	<b>Channel3 Status</b> This bit is the result of an OR operation of the various status bits of channel 3 (see STS register).
<b>CH4</b>	4	rh	<b>Channel4 Status</b> This bit is the result of an OR operation of the various status bits of channel 4 (see STS register).
<b>CH5</b>	5	rh	<b>Channel5 Status</b> This bit is the result of an OR operation of the various status bits of channel 5 (see STS register).
<b>CH6</b>	6	rh	<b>Channel6 Status</b> This bit is the result of an OR operation of the various status bits of channel 6 (see STS register).

**Flexible CRC Engine (FCE)**

Field	Bits	Type	Description
<b>CH7</b>	7	rh	<b>Channel7 Status</b> This bit is the result of an OR operation of the various status bits of channel 7 (see STS register).
<b>0</b>	31:8	r	<b>Reserved</b> Read as 0. Should be written with 0.

## Flexible CRC Engine (FCE)

### 17.4.3 CRC Channel Control/Status Registers

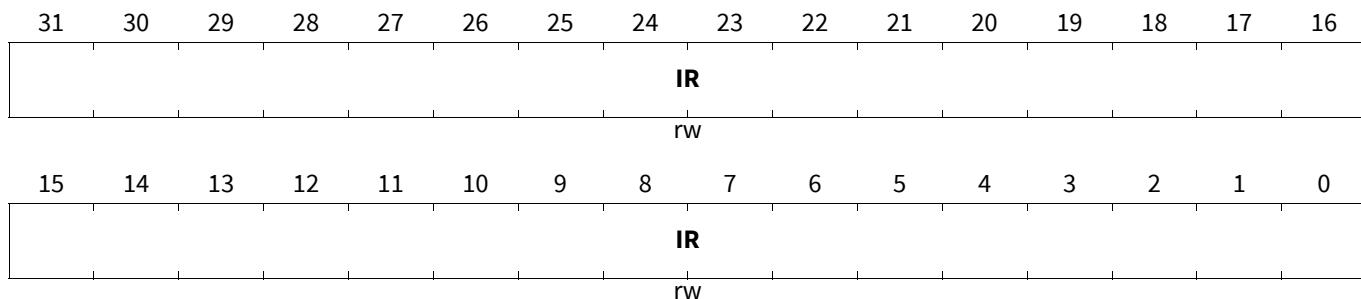
#### Input Register i

**IR<sub>i</sub> (i=0-7)**

#### Input Register i

(100<sub>H</sub>+i\*20<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
IR	31:0	rw	<b>Input Register</b> This bit field holds the input data to be computed. In case the channel is configured to use 16-bit or 8-bit CRC, only the LSB 16 or 8-bits will be used as input.

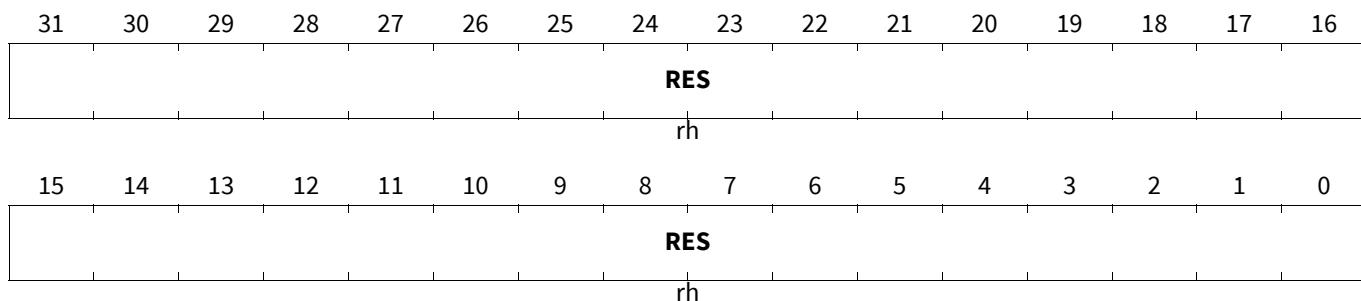
#### CRC Result Register i

**RES<sub>i</sub> (i=0-7)**

#### CRC Result Register i

(104<sub>H</sub>+i\*20<sub>H</sub>)

Application Reset Value: FFFF FFFF<sub>H</sub>



Field	Bits	Type	Description
RES	31:0	rh	<b>Result Register</b> Returns the final CRC value including CRC reflection and final XOR according to the CFG register configuration. Writing to this register produces a bus error. If the channel is configured to use 16-bit or 8-bit CRC, the MSB 16 or 24 bits respectively shall be read as 0.

## Flexible CRC Engine (FCE)

## CRC Configuration Register i

CFG*i* (*i*=0-7)

## CRC Configuration Register i

(108<sub>H</sub>+*i*\*20<sub>H</sub>)Application Reset Value: 0000 0700<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												KERNEL			
r												rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				BYTES WAP	XSEL	REFO UT	REFIN	0		ALR	CCE	BEI	LEI	CEI	CMI
r				rw	rw	rw	rw	r		rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
<b>CMI</b>	0	rw	<b>CRC Mismatch Interrupt</b> 0 <sub>B</sub> CRC Mismatch Interrupt is disabled 1 <sub>B</sub> CRC Mismatch Interrupt is enabled
<b>CEI</b>	1	rw	<b>Configuration Error Interrupt</b> When enabled, a Configuration Error Interrupt is generated whenever a mismatch is detected in the CFG and CHECK redundant registers. 0 <sub>B</sub> Configuration Error Interrupt is disabled 1 <sub>B</sub> Configuration Error Interrupt is enabled
<b>LEI</b>	2	rw	<b>Length Error Interrupt</b> When enabled, a Length Error Interrupt is generated if software writes to IR register with LENGTH equal to 0 and CFG.CCE is set to 1. 0 <sub>B</sub> Length Error Interrupt is disabled 1 <sub>B</sub> Length Error Interrupt is enabled
<b>BEI</b>	3	rw	<b>Bus Error Interrupt</b> When enabled, an interrupt (BEF) is generated if a bus write transaction with an access width smaller than the kernel width is issued to the input register. In this case, the corresponding value written to the IR is discarded and no CRC computation takes place. 0 <sub>B</sub> Bus Error Interrupt is disabled 1 <sub>B</sub> Bus Error Interrupt is enabled
<b>CCE</b>	4	rw	<b>CRC Check Comparison</b> 0 <sub>B</sub> CRC check comparison at the end of a message is disabled 1 <sub>B</sub> CRC check comparison at the end of a message is enabled. In this case, if length error is set (STS.LEF = 1) and IR is written to, then length is set to all ones, independent of the previous length value. The ALR bit is also ignored in such a condition.
<b>ALR</b>	5	rw	<b>Automatic Length Reload</b> 0 <sub>B</sub> Disables automatic reload of the LENGTH field. 1 <sub>B</sub> Enables automatic reload of the LENGTH field at the end of a message.

## Flexible CRC Engine (FCE)

Field	Bits	Type	Description
<b>REFIN</b>	8	rw	<b>IR Byte Wise Reflection</b> 0 <sub>B</sub> IR Byte Wise Reflection is disabled 1 <sub>B</sub> IR Byte Wise Reflection is enabled
<b>REFOUT</b>	9	rw	<b>CRC Bit Wise Reflection</b> The alignment of the reflection is the same as the kernel polynomial width. Eg. 32-bit kernel: Bitwise reflection by 32-bits. 0 <sub>B</sub> CRC-bit wise is disabled 1 <sub>B</sub> CRC-bit wise is enabled
<b>XSEL</b>	10	rw	<b>Selects the value to be xored with the final CRC</b> 0 <sub>B</sub> 0x00000000 1 <sub>B</sub> 0xFFFFFFFF
<b>BYTESWAP</b>	11	rw	<b>Swaps the order of the bytes in the IR input register.</b> 0 <sub>B</sub> The order of bytes in IR register are not swapped before CRC computation. 1 <sub>B</sub> The order of bytes in the IR register are swapped before CRC computation. Big-endian input is converted to Little-endian and vice versa. (When 8-bit CRC is chosen, this has no effect).
<b>KERNEL</b>	19:16	rw	<b>Selects the CRC Kernel (Polynomial Engine) used by this channel.</b> Other possible values are reserved for additional kernels that may be added in the future. If these values are used, then KERNEL3 is used by default. 0 <sub>H</sub> Kernel 0 is used. 1 <sub>H</sub> Kernel 1 is used. 2 <sub>H</sub> Kernel 2 is used. 3 <sub>H</sub> Kernel 3 is used.
<b>0</b>	7:6, 15:12, 31:20	r	<b>Reserved</b> Read as 0; should be written with 0.

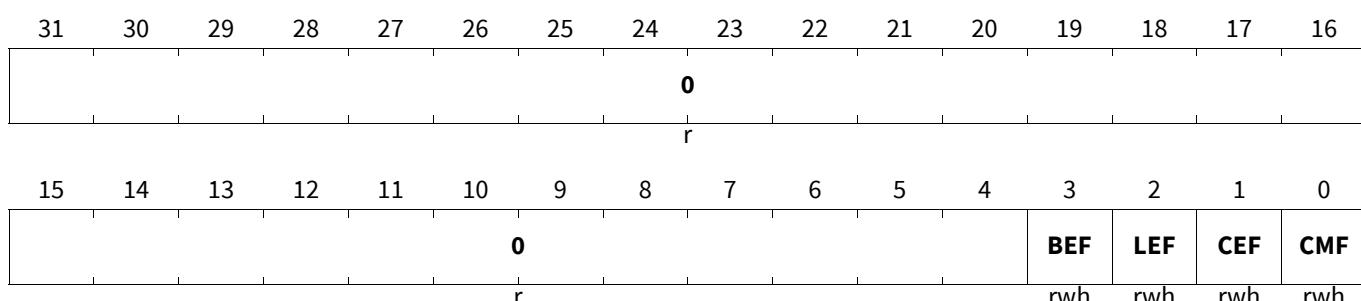
## CRC Status Register i

### STS<sub>i</sub> (i=0-7)

### CRC Status Register i

(10C<sub>H</sub>+i\*20<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



## Flexible CRC Engine (FCE)

Field	Bits	Type	Description
<b>CMF</b>	0	rwh	<b>CRC Mismatch Flag</b> This bit is set per hardware only. To clear this bit, software must write a 0 to this bit field location. Writing 1 to this bit has no effect.
<b>CEF</b>	1	rwh	<b>Configuration Error Flag</b> This bit is set per hardware only. To clear this bit, software must write a 0 to this bit field location. Writing a 1 has no effect.
<b>LEF</b>	2	rwh	<b>Length Error Flag</b> This bit is set per hardware only. To clear this bit, software must write a 0 to this bit field location. Writing 1 has no effect.
<b>BEF</b>	3	rwh	<b>Bus Error Flag</b> This bit is set per hardware only. To clear this bit, software must write a 0 to this bit field location. Writing 1 has no effect.
<b>0</b>	31:4	r	<b>Reserved</b> Read as 0; should be written with 0.

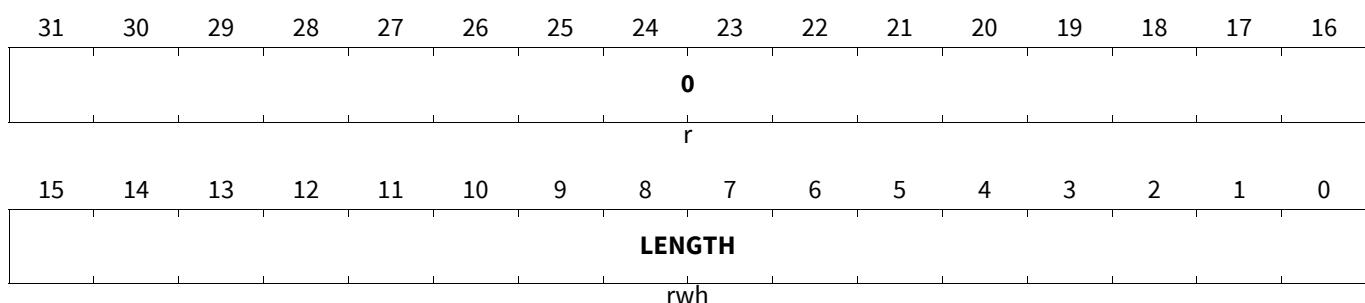
### CRC Length Register i

**LENGTH<sub>i</sub> (i=0-7)**

**CRC Length Register i**

(110<sub>H</sub>+i\*20<sub>H</sub>)

**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>LENGTH</b>	15:0	rwh	<b>Message Length Register</b> Number of words (bit width of each word in terms of KERNEL polynomial width) building the message over which the CRC checksum is calculated. This bit field is modified by the hardware: every write to the IR register decrements the value of the LENGTH bit field. If the CFG.ALR field is set to 1, the LENGTH field shall be reloaded with its configuration value at the end of the cycle where LENGTH reaches 0.
<b>0</b>	31:16	r	<b>Reserved</b> Read as 0; should be written with 0.

## Flexible CRC Engine (FCE)

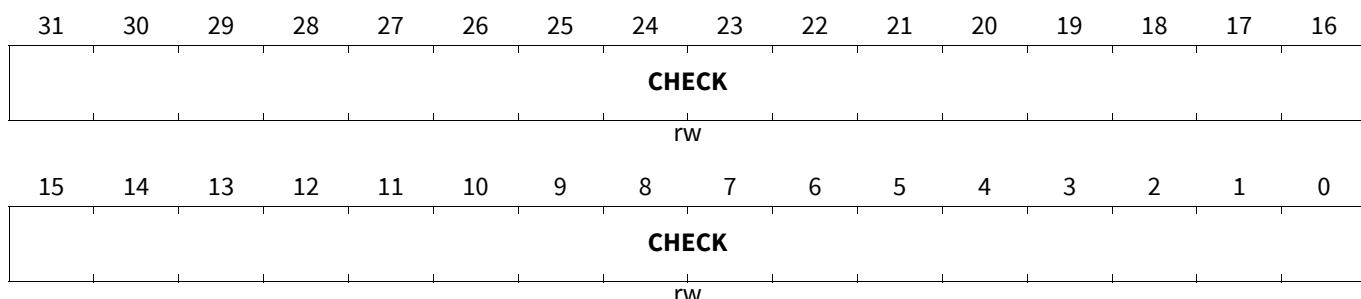
### CRC Check Register i

**CHECK<sub>i</sub> (i=0-7)**

**CRC Check Register i**

**(114<sub>H</sub>+i\*20<sub>H</sub>)**

**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>CHECK</b>	31:0	rw	<b>CHECK Register</b> Expected CRC value to be checked by the hardware upon detection of a 1 to 0 transition of the LENGTH register. The comparison is enabled by the CFG.CCE bit field

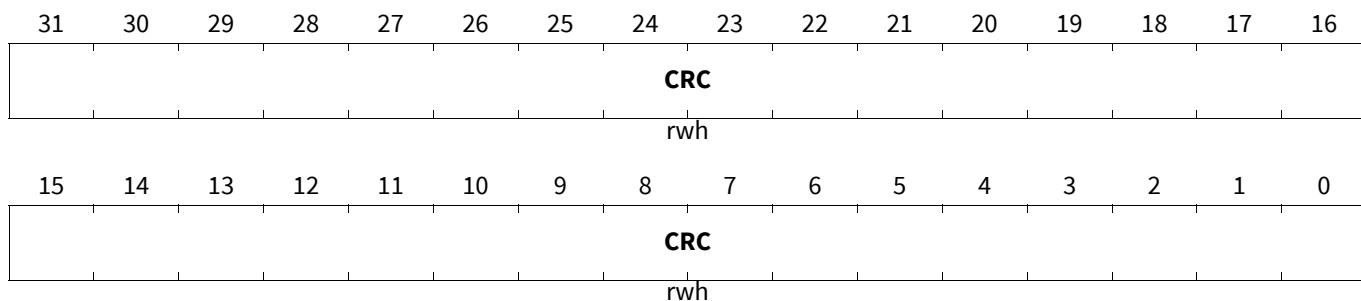
### CRC Register i

**CRC<sub>i</sub> (i=0-7)**

**CRC Register i**

**(118<sub>H</sub>+i\*20<sub>H</sub>)**

**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>CRC</b>	31:0	rwh	<b>CRC Register</b> This register enables to directly access the internal CRC register

## Flexible CRC Engine (FCE)

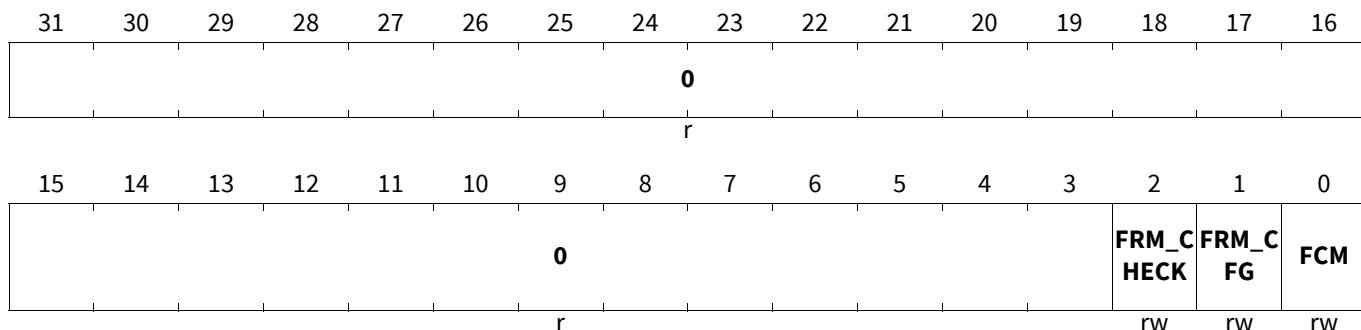
### CRC Test Register i

**CTR<sub>i</sub> (i=0-7)**

**CRC Test Register i**

(11C<sub>H</sub>+i\*20<sub>H</sub>)

**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>FCM</b>	0	rw	<b>Force CRC Mismatch</b> Forces the CRC compare logic to issue an error regardless of the CHECK and CRC values. The hardware detects a 0 to 1 transition of this bit field and triggers a CRC Mismatch interrupt
<b>FRM_CFG</b>	1	rw	<b>Force CFG Register Mismatch</b> This field is used to control the error injection mechanism used to check the compare logic of the redundant CFG registers. This is a one shot operation. When the hardware detects a 0 to 1 transition of this bit field it triggers a Configuration Mismatch interrupt (if enabled by the corresponding CFGm register).
<b>FRM_CHECK</b>	2	rw	<b>Force Check Register Mismatch</b> This field is used to control the error injection mechanism used to check the compare logic of the redundant CHECK registers. This is a one shot operation. The hardware detects a 0 to 1 transition of this bit field and triggers a Check Register Mismatch interrupt (if enabled by the corresponding CFGm register).
<b>0</b>	31:3	r	<b>Reserved</b> Read as 0; should be written with 0.

### 17.5 Debug

The FCE has no specific debug feature.

## Flexible CRC Engine (FCE)

### 17.6 IO Interfaces

**Table 565 List of FCE Interface Signals**

Interface Signals	I/O	Description
SRC_FCE	out	<b>FCE Service Request</b>

### 17.7 Revision History

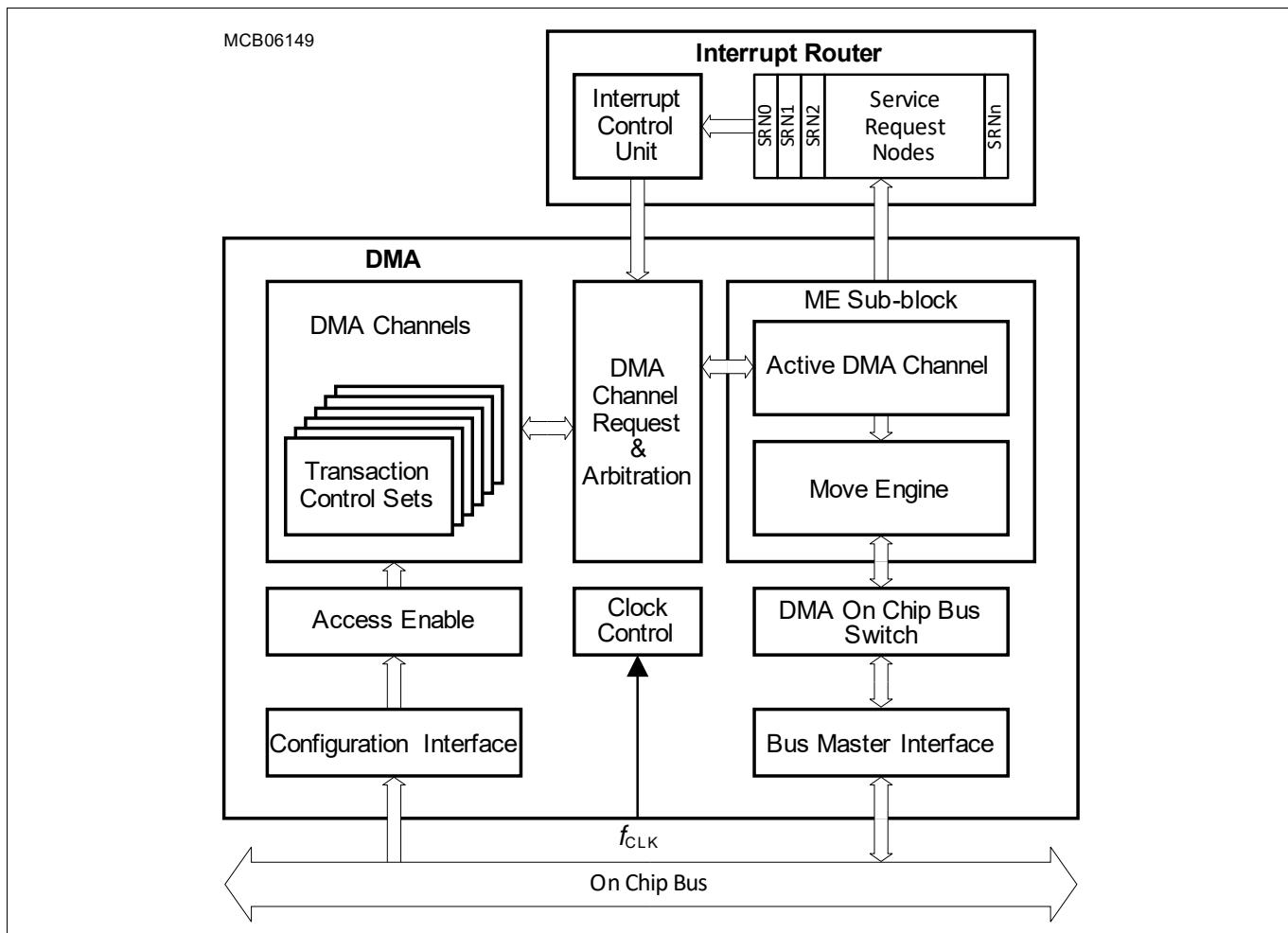
**Table 566 Revision History**

Reference	Change to Previous Version	Comment
V4.2.9		
<a href="#">Page 11</a>	Removed from <a href="#">Figure 185</a> , the “Property: Redundant Register shall be physically isolated from the functional Register”	

## Direct Memory Access (DMA)

### 18 Direct Memory Access (DMA)

The DMA (at [Figure 189](#)) shall move data from a source module to a destination module without CPU intervention.



**Figure 189 Block Diagram of DMA**

#### DMA Glossary

**Table 567 DMA Acronyms**

Acronym	Description
ACLL	Accumulated Linked List
CONLL	Conditional Linked List
CH	Channel
DER	Destination Error
DLER	Linked List Error
DMA	Direct Memory Access
DMALL	Direct Memory Access Linked List
DMARAM	Direct Memory Access Random Access Memory
EH	Error Handler

## Direct Memory Access (DMA)

**Table 567 DMA Acronyms (cont'd)**

Acronym	Description
ME	Move Engine
PATDET	Pattern Detection
RAMER	RAM Error
RDCRC	Read Data Cyclic Redundancy Check
RP	Resource Partition
RROAT	Reset Request Only After Transaction
SAFLL	Safe Linked List
SDCRC	Source Destination Cyclic Redundancy Check
SER	Source Error
SLLER	Safe Linked List Error
TCS	Transaction Control Set
TRL	Transaction/Transfer Request Lost
TS	Trigger Set

**Table 568 DMA Terms**

Term	Description
DMA Configuration Data	The DMA configuration data configures the DMA configuration structure.
DMA Configuration Structure	The DMA configuration structure is used to configure CH, RP and ME. The DMA configuration structure is defined by the DMA address map.  Note: <i>The DMA configuration structure does not include IR, SMU, etc. which is needed to make the DMA function within the system.</i>  3. <i>The DMA configuration structure and DMA configuration data defines all DMA move functions.</i>
DMA Software Request	The CPU initiates a DMA transfer or a DMA transaction in a DMA channel.
DMA Hardware Request	The Interrupt Router initiates a DMA transfer or a DMA transaction in a DMA channel.
DMA Daisy Chain Request	As soon as the current DMA transaction completes, the DMA channel initiates a DMA transfer or a DMA transaction in the next lower priority DMA channel.
DMA Auto Start Request	As soon as the current DMA transaction completes, a linked list operation loads a new TCS and initiates a DMA transfer or a DMA transaction.
DMA Request	DMA software request, DMA hardware request, DMA daisy chain request or DMA auto start request.
DMA Read Move Data	Data read from the source module.
DMA Write Move Data	Data written to the destination module.
DMA Channel Interrupt Triggers	DMA signalling of the successful completion of a DMA move function.
DMA RP Error Interrupt Triggers	DMA signalling of a DMA move function error.
DMA Alarms	DMA signalling of a DMA safety error.

## Direct Memory Access (DMA)

**Table 568 DMA Terms (cont'd)**

Term	Description
DMA Address Checksum	Cyclic redundancy checksum calculated according to the IEEE 802.3 standard for addresses generated during a DMA transaction.
DMA Data Checksum	Cyclic redundancy checksum calculated according to the IEEE 802.3 standard for data moved during a DMA transaction.
DMA Timestamp	Appendage of timestamp to a DMA transaction.

The terms (at [Table 569](#)) are used to define the structure of the data move function.

**Table 569 Structure of Data Move Function**

Term	Description
DMA Move	A DMA move is an operation that always consists of two parts: 1. A DMA read move that loads DMA read move data from a source module to the DMA. 2. A DMA write move that stores DMA write move data from the DMA to a destination module.
DMA Transfer	A DMA transfer shall be composed of 1, 2, 3, 4, 5, 8, 9 or 16 DMA moves.
DMA Transaction	A DMA transaction shall be composed of at least one DMA transfer.
Linked List	A linked list is a series of DMA transactions executed in the same DMA channel.

## 18.1 Feature List

The DMA is a fast and flexible DMA controller that has the following features:

- **Resource Partitions**
  - An application running one set of defined move data functions shall be free from interference by another application running another set of defined move data functions.
  - Each RP has independent **Access Enable** control via enabling individual Master TAG identifiers to have write access enable to the RP and assigned DMA channels.
  - Each RP has a unique master tag identifier driven onto the on chip bus during a DMA move.
  - Each RP executes on chip bus accesses in supervisor or user mode.
- **DMA Channels**
  - The DMA supports multiple independent DMA channels.
  - Each DMA channel shall be assigned to a RP.
  - Each DMA channel shall be individually programmable.
  - Each DMA channel TCS shall be stored in DMARAM.
  - The DMA channel source and destination address pointers shall be 32-bit wide address counters.
  - Wrap buffer addressing mode with flexible circular buffer sizes.
  - The DMA channel source and destination wrap buffers shall be selectable.
  - Programmable data width of DMA moves.
- **Double Buffering Operations**
  - The DMA transaction can execute read or fill DMA transfers from one of two source or destination buffers.
  - A control bit allows the re-direction of DMA transfers from the one buffer to the other buffer.
- **DMA Linked List (DMALL)**
  - The current DMA transaction can load the next DMA channel TCS into the DMARAM by overwriting the existing DMA channel TCS.

## Direct Memory Access (DMA)

- The next DMA transaction may be auto started.
- **DMA Channel Request Control**
  - **DMA Software Request**
  - **DMA Hardware Request**
  - **DMA Daisy Chain Request**
  - **DMA Auto Start Request**
- **Move Engine**
  - Any ME shall service a DMA request from any DMA channel.
  - DMA requests from the highest number DMA channel are serviced first by a ME.
  - Multiple MEs support the parallel servicing of DMA requests.
  - SRI-source to SRI-destination data block move throughput <8 Mbyte DMA moves per DMA transaction.
  - SPB-source to SPB-destination data block move throughput <1 Mbyte DMA moves per DMA transaction.
- **DMA On Chip Bus Switch**
  - DMA read moves and DMA write moves are directed by the DMA on chip bus switch to different sources and destinations depending on the source or destination address.
  - Buffer capability for move actions on the buses (at least 1 x DMA move per bus is buffered).
- Interrupt Triggers
  - Each DMA channel generates one interrupt trigger with an unique interrupt vector and priority level.
  - Each DMA RP generates one error interrupt trigger with an unique interrupt vector and priority level.
- Operating frequencies
  - The DMA configuration and request control function works at the SPB clock frequency.
  - The ME function works at the SRI clock frequency in order to maximise the data throughput for DMA moves from SRI source addresses to SRI destination addresses.

### 18.2 Overview

The DMA moves data from source locations to destination locations without the intervention of the CPU or other on chip devices. A data move is controlled by the TCS of an active DMA channel executed by a ME. A DMA channel is activated by a DMA request.

---

## Direct Memory Access (DMA)

### 18.3 Functional Description

#### 18.3.1 Configuration Interface

The DMA implements a standard FPI slave interface compliant with the FPI bus protocol on the SPB bus. The DMA configuration interface supports single data transfers and does not support block transfers.

#### 18.3.2 Resource Partitions

During DMA configuration, each DMA channel shall be assigned to a RP.

##### 18.3.2.1 Access Enable

Each RP has its own access enable protection. A slave destination controls access to its bus peripheral interfaces and kernel address space. Each on chip resource with bus master capability has a unique master tag identifier that is used to identify the source of an on chip bus transaction. The master tag identifier based access protection is used to enable write accesses to individual slave address ranges.

##### 18.3.2.2 DMA Moves

Each RP has a unique master tag identifier driven onto the on chip bus during a DMA read move or DMA write move. Mode control selects if a DMA move on chip bus access is made in supervisor mode or user mode.

*Note:* See *On-Chip System Connectivity chapter* for on chip bus master TAG identifier assignments.

##### 18.3.2.3 DMA RP Error Interrupt Service Request

Each RP generates one error interrupt service request to cover all error events for DMA channels assigned to that RP including servicing of DMA requests by the ME:

- DMA channel **TRL** interrupt service request.
- ME SER and DER error interrupt service request.
- ME **DMARAM Integrity Error** error interrupt service request.
- ME **Linked List Operation TCS Load Error** error interrupt service request.
- ME **SAFLL DMA Address Checksum Error** error interrupt service request.

If a DMA RP error interrupt service request is triggered then the software RP application EH shall read the contents of the error status registers to identify the origin of the error.

### 18.3.3 DMA Channels

Each DMA channel is assigned to a RP and stores the context of an independent DMA operation.

#### 18.3.3.1 DMA Channel Request Control

The DMA channel request control (at **Figure 190**) is implemented for each DMA channel.

The DMA channel operation is individually programmable. The following types of DMA requests are possible:

- **DMA Software Request** initiated by a CPU.
- **DMA Hardware Request** initiated by the Interrupt Router (IR) Interrupt Control Unit (ICU).
- **DMA Daisy Chain Request** initiated by the next higher priority DMA channel.

## Direct Memory Access (DMA)

- **DMA Auto Start Request** initiated by the loading of the next TCS during a **DMA Linked List (DMALL)**, **Accumulated Linked List (ACLLL)**, **Safe Linked List (SAFLL)** or **Conditional Linked List (CONLL)** operation.

The DMA channel status flag TSR.CH indicates if a DMA request is pending. TSR.CH may be cleared at the start of a DMA transfer or at the end of a DMA transaction. It follows that a DMA request may trigger a single **DMA Transfer** or one complete **DMA Transaction**.

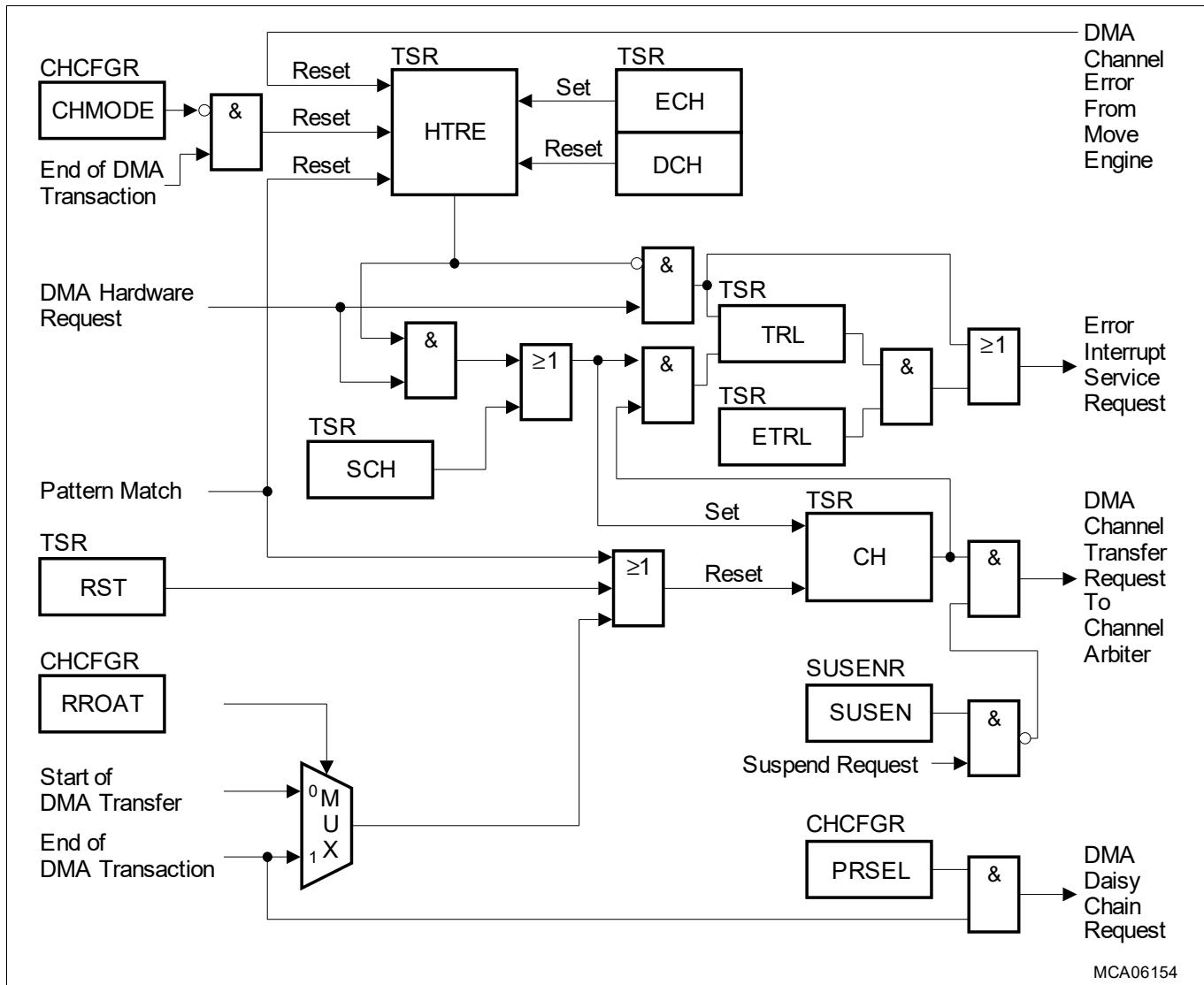


Figure 190 DMA Channel Request Control

### 18.3.3.1.1 DMA Channel States

Table 570 DMA Channel States

State	Status	Comments
Idle State	DMA channel TSR.CH = 0 <sub>B</sub>	No <b>DMA Request</b> is pending.
<b>Reset State</b>	DMA channel TSR.CH = 0 <sub>B</sub>	No <b>DMA Request</b> is pending. DMA channel TCS bits cleared.
<b>Halt State</b>	DMA channel TSR.HLTACK = 1 <sub>B</sub>	A <b>DMA Request</b> shall not be serviced by a ME.
Pending State	DMA channel TSR.CH = 1 <sub>B</sub>	<b>DMA Request</b> is waiting to be serviced by a ME.
Active State		ME is servicing a <b>DMA Request</b> .

## Direct Memory Access (DMA)

### 18.3.3.1.2 Reset Request Only After Transaction (RROAT)

The clearing of the DMA channel TSR.CH bit is controlled as follows:

- CHCFGR.RROAT = 0<sub>B</sub>
  - DMA channel TSR.CH is cleared at the start of each DMA transfer.
  - One **DMA Request** starts one single **DMA Transfer**.
- CHCFGR.RROAT = 1<sub>B</sub>
  - DMA channel TSR.CH is cleared at the end of each DMA transaction.
  - One **DMA Request** starts one complete **DMA Transaction**.

### 18.3.3.2 DMA Software Request

One **DMA Software Request** may start one complete DMA transaction or one single DMA transfer.

If the DMA channel is triggered only by software then a **DMA Hardware Request** should be disabled.

Software must set TSR.DCH to 1<sub>B</sub> to disable a **DMA Hardware Request** (TSR.HTRE = 0<sub>B</sub>).

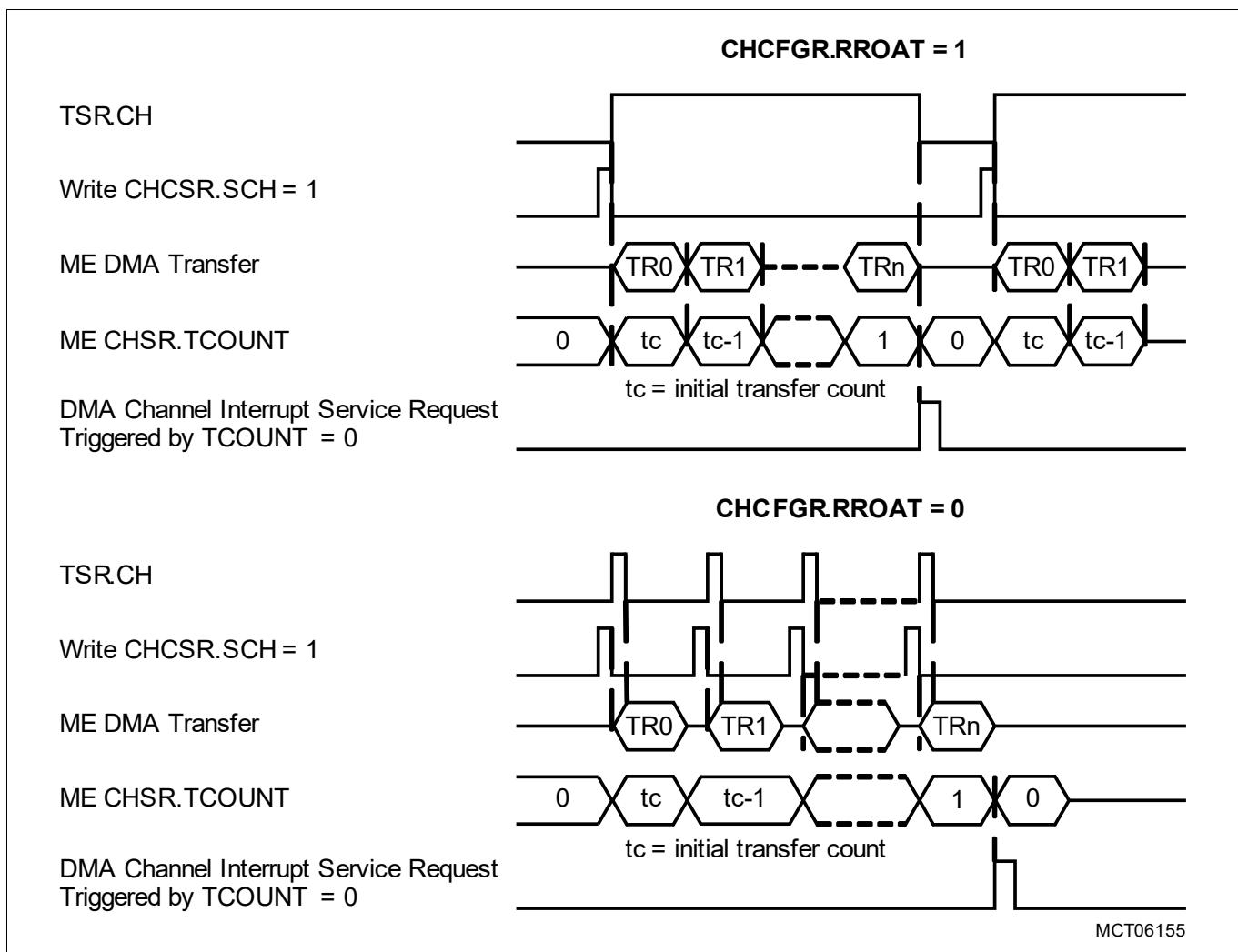


Figure 191 Software Control

The following DMA channel configuration is required to initiate one complete **DMA Transaction** under software control:

- DMA channel CHCFGR.RROAT = 1<sub>B</sub>

## Direct Memory Access (DMA)

A **DMA Software Request** is initiated by setting the DMA channel CHCSR.SCH to  $1_B$  with the result that TSR.CH is set to  $1_B$ . At the start of the DMA transaction, the value of the DMA channel reload value CHCFG.RTREL is loaded into ME CHSR.TCOUNT and DMA transfers are executed. After each DMA transfer, ME CHSR.TCOUNT is decremented and the next source and destination addresses are calculated. When TCOUNT decrements to  $0_D$ , the DMA channel status flag TSR.CH is reset. Setting the DMA channel CHCSR.SCH =  $1_B$  again starts a new DMA transaction of the DMA channel with the stored TCS parameters.

The following DMA channel configuration is required to initiate each single DMA Transfer under software control:

- DMA channel CHCFG.RRQAT =  $0_B$

A **DMA Software Request** must be initiated for each DMA transfer by setting the DMA channel CHCSR.SCH to  $1_B$ .

### 18.3.3.3 DMA Hardware Request

A **DMA Hardware Request** is enabled/disabled by the DMA channel Hardware Transaction/Transfer Request Enable (HTRE) bit TSR.HTRE. The HTRE functionality is as follows:

- Software may set (TSR.ECH =  $1_B$ ) or clear (TSR.DCH =  $1_B$ ) DMA channel TSR.HTRE.
- Cleared as a result of the ME reporting an error for the DMA channel.
- Single Mode: at the start of the last DMA transfer of a DMA transaction.

**Table 571 Conditions to Set/Reset DMA channel TSR.HTRE**

DMA channel TSR.ECH	DMA channel TSR.DCH	Start of last DMA transfer of a DMA transaction <sup>1)</sup>	Modification of DMA channel TSR.HTRE
0	0	0	Unchanged
1	0	0	Set
X	1	X	Reset
X	X	1	Reset

1) In **Single Mode** only. In **Continuous Mode**, the end of a DMA transaction has no impact.

### DMA Channel Mode

The DMA channel mode bit CHCFG.CHMODE controls the DMA channel mode as follows:

- **Single Mode** (DMA channel CHCFG.CHMODE =  $0_B$ )
  - **DMA Hardware Request** is disabled by hardware on completion of a DMA transaction.
- **Continuous Mode** (DMA channel CHCFG.CHMODE =  $1_B$ )
  - **DMA Hardware Request** is not disabled by hardware on completion of a DMA transaction.

### Single Mode

The following DMA channel configuration is required to initiate one complete DMA Transaction under hardware control in Single Mode:

- DMA channel CHCFG.CHMODE =  $0_B$
- DMA channel CHCFG.RRQAT =  $1_B$
- DMA channel CHCFG.PRSEL =  $0_B$
- DMA channel TSR.ECH =  $1_B$

Setting DMA channel TSR.ECH to  $1_B$  enables a **DMA Hardware Request** (TSR.HTRE =  $1_B$ ) for the DMA channel. When the ICU generates a **DMA Hardware Request**, TSR.CH is set high. If the DMA channel wins channel arbitration then the DMA channel transitions to the active state. The value of CHCFG.RTREL is loaded into the ME CHSR.TCOUNT and the DMA transaction is started by executing the first DMA transfer. After each DMA transfer, ME

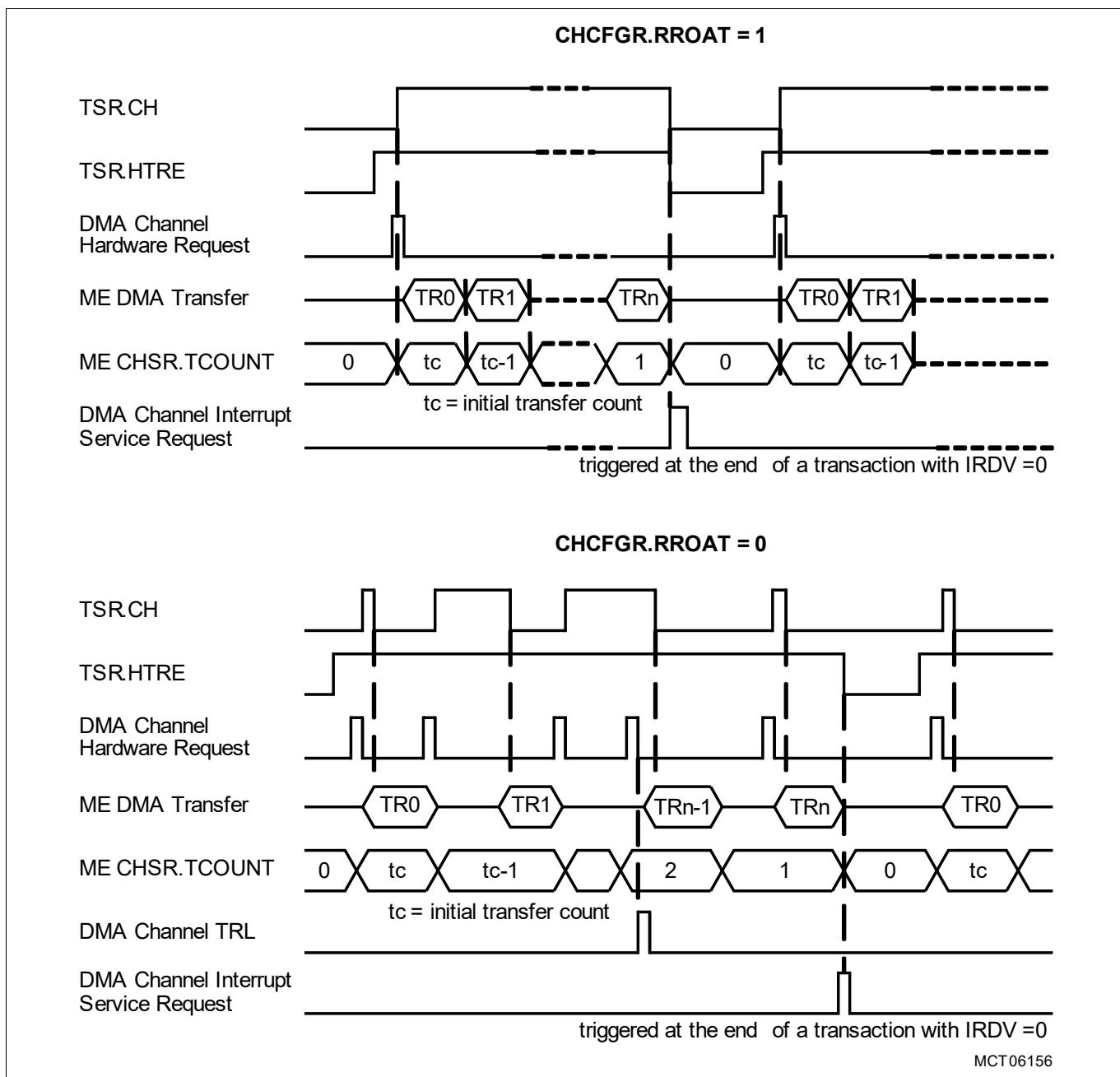
## **Direct Memory Access (DMA)**

CHSR.TCOUNT is decremented and the next source and destination addresses are calculated. When TCOUNT decrements to 0<sub>B</sub>, **DMA Hardware Request** is disabled and status flags TSR.CH and TSR.HTRE are reset. In order to start a new hardware-controlled DMA transaction, a **DMA Hardware Request** must be enabled again by software writing TSR.ECH = 1<sub>B</sub> to set TSR.HTRE. The hardware request disable function in Single Mode is typically needed to reprogram a DMARAM channel TCS before the next **DMA Hardware Request** initiates a DMA transaction.

The following DMA channel configuration is required to initiate each single **DMA Transfer** under hardware control in Single Mode:

- DMA channel CHCFG.RROAT = 0<sub>B</sub>

In this DMA channel configuration, TSR.CH is cleared at the start of each DMA transfer and a new **DMA Hardware Request** must be generated to start the next DMA transfer.



Downloaded by IFXDMZ\par-guowill 09/06/2020 03:48:39

**Figure 192 Hardware Control in Single Mode**

## Direct Memory Access (DMA)

### Continuous Mode

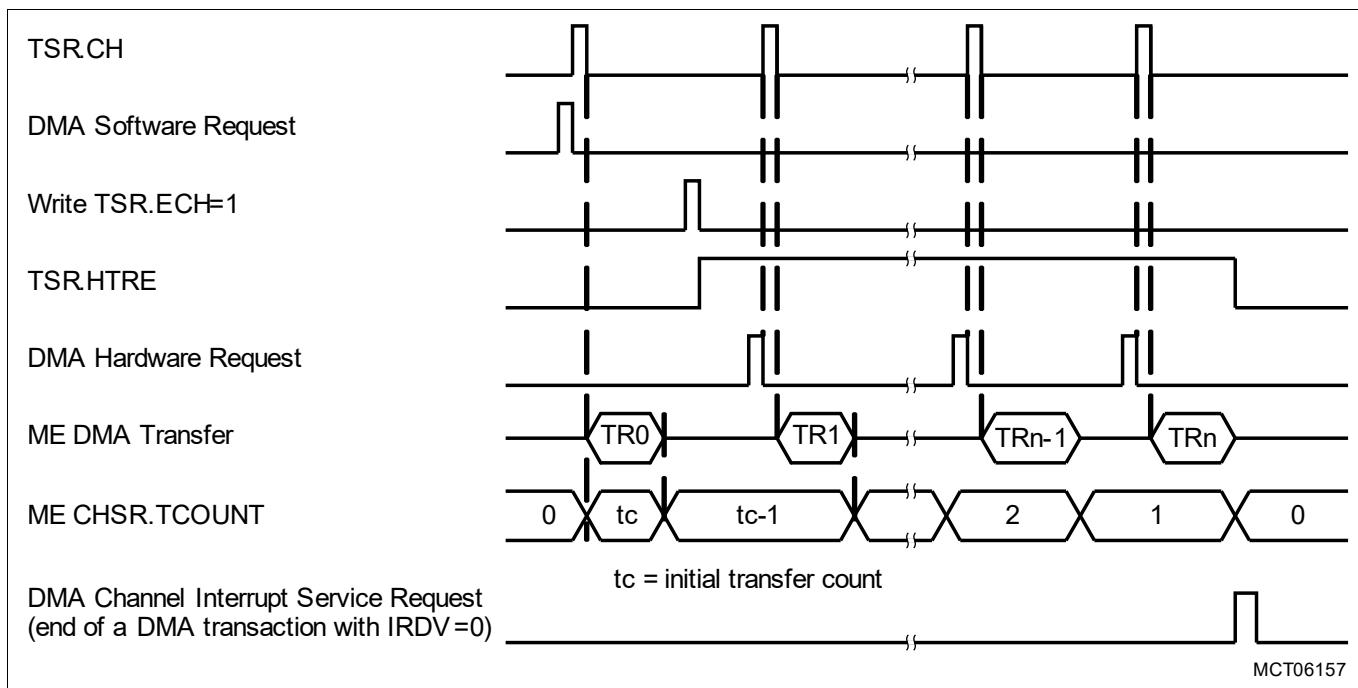
If the DMA channel is configured for Continuous Mode (CHCFGR.CHMODE = 1<sub>B</sub>), then TSR.HTRE is not reset at the end of a DMA transaction. On completion of the current DMA transaction (ME CHSR.TCOUNT = 0<sub>D</sub>) each new **DMA Hardware Request** will start a new DMA transaction with the stored DMA channel TCS.

### 18.3.3.4 Combined DMA Software Request and DMA Hardware Request

A DMA channel may be operated under combined software and hardware control. The example (at **Figure 193**) demonstrates combined control:

- The first DMA transfer is triggered by a **DMA Software Request** setting the DMA channel CHCSR.SCH = 1<sub>B</sub>.
- A **DMA Hardware Request** is still disabled i.e. DMA channel TSR.HTRE = 0<sub>B</sub>.
- Software enables a **DMA Hardware Request** by setting the DMA channel TSR.ECH = 1<sub>B</sub>.
- Each subsequent DMA transfer is triggered by a **DMA Hardware Request** from the ICU.

In the example, the DMA channel operates in Single Mode (DMA channel CHCFGR.CHMODE = 0<sub>B</sub>). In single mode, the DMA channel TSR.HTRE is reset by hardware when ME CHSR.TCOUNT = 0<sub>D</sub> at the end of the DMA transaction.



**Figure 193 Transaction Start by Software, Continuation by Hardware**

### Transaction Request Lost (TRL)

When a DMA channel is configured to be triggered by parallel hardware and software requests, if a **DMA Software Request** and a **DMA Hardware Request** collide in the same clock cycle then a **TRL** event shall be flagged.

### 18.3.3.5 DMA Daisy Chain Request

DMA channels shall be configured for DMA daisy chain request by setting DMA channel CHCFGR.PRSEL.

When a higher priority DMA channel completes a DMA transaction it will initiate a DMA transaction in the next lower priority DMA channel by setting the access pending bit TSR.CH bit. **DMA Daisy Chain Request** is limited to a higher priority DMA channel initiating a DMA request in the next lower priority DMA Channel.

Enabling the daisy chain disables the **DMA Channel Interrupt Service Request** trigger in the next higher priority DMA channel. In a typical DMA daisy chain application only the lowest priority DMA channel is required to

## Direct Memory Access (DMA)

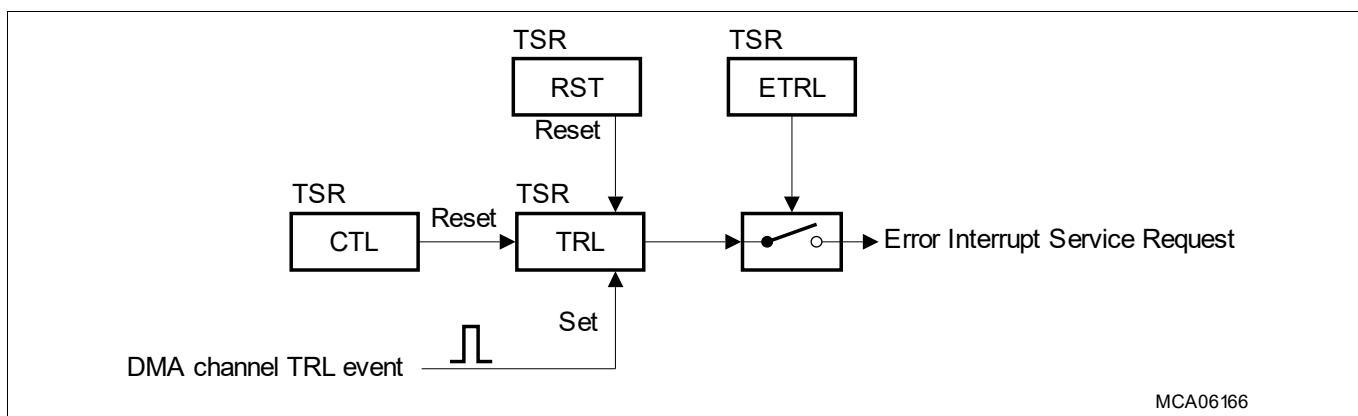
generate a **DMA Channel Interrupt Service Request**. When the sequence of DMA transactions from the highest to lowest priority DMA channel has completed then the lowest priority DMA channel in the daisy chain will generate a **DMA Channel Interrupt Service Request** to signal the end of the DMA operation.

If DMA channels are configured in a daisy chain, a DMA transfer or DMA transaction in the highest priority DMA channel is initiated by a **DMA Software Request** or a **DMA Hardware Request**. DMA transactions in the lower priority DMA channels are triggered by a **DMA Daisy Chain Request** in order to improve DMA latency.

### 18.3.3.6 DMA Channel Transaction Request Lost Interrupt Service Request

A DMA channel **TRL** event occurs for the following conditions:

- If a **DMA Request** is detected and the DMA channel TSR.CH is set, the DMA channel shall set the DMA channel TRL bit ( $TSR.TRL = 1_B$ ). If the DMA channel enable TRL bit is set ( $TSR.ETRL = 1_B$ ), the DMA shall trigger a **DMA RP Error Interrupt Service Request** (at **Figure 194**).
- If the DMA channel is disabled for hardware requests and a **DMA Hardware Request** is detected, the DMA shall set the DMA channel TRL bit ( $TSR.TRL = 1_B$ ) and trigger a **DMA RP Error Interrupt Service Request**.



**Figure 194 DMA Channel Transaction Request Lost Interrupt Service Request**

An **Error Handler** shall interrogate the DMA channels to identify the source of the error. Software may clear TSR.TRL by setting the DMA channel TSR.CTL or TSR.RST.

### 18.3.3.7 DMA Service Requests

Interrupt Requests are prioritized by the Interrupt Router and processed by one of the Service Providers (CPU or DMA). The DMA interfaces to an Interrupt Control Unit (ICU) instantiated in the Interrupt Router (IR).

DMA channels are associated with the Service Request Priority Number (SRPN) bit field programmed in the Service Request Control (SRC) Register SRC.SRPN. For example:

- DMA channel 000 equates to  $SRC.SRPN = 0_D$  programmed in IR.
- DMA channel 001 equates to  $SRC.SRPN = 1_D$  programmed in IR.
- DMA channel 002 equates to  $SRC.SRPN = 2_D$  programmed in IR.
- DMA channel 003 equates to  $SRC.SRPN = 3_D$  programmed in IR.
- DMA channel 004 equates to  $SRC.SRPN = 4_D$  programmed in IR.

The routing of a hardware service request to a service provider destination is determined by the IR Type Of Service (TOS) control bit field SRC.TOS. The DMA will acknowledge all service requests. If the value programmed in the SRC.SRPN is for an invalid DMA channel then the DMA will take no action. The user must programme valid SRC.SRPN values for the DMA.

## Direct Memory Access (DMA)

### 18.3.3.8 DMA Request Arbitration

The DMA arbiter continuously monitors all DMA channels for a pending **DMA Request** when the DMA channel is not in the halt state ( $\text{TSR.HLTREQ} = 1_B$ ) and/or the suspend state ( $\text{SUSASR.SUSAC} = 1_B$ ).

The highest number DMA channel with a pending **DMA Request** wins the DMA channel arbitration. The pending DMA request is forwarded to the highest number available ME. The ME reads the DMA channel TCS from the DMARAM and loads the TCS into the ME active channel register set and executes a DMA transfer. On completion of each DMA transfer the DMA performs an **Arbitration Sequence**.

#### Arbitration Sequence

- **Rule 1:** If there is a higher priority DMA channel with a pending **DMA Request** then
  - **Rule 1.1:** The ME shall write back the active DMA channel updated TCS to the DMARAM.
  - **Rule 1.2:** The ME reads the higher priority DMA channel TCS from the DMARAM to the ME.
  - **Rule 1.3:** The ME shall execute a DMA transfer.
  - **Rule 1.4:** On completion of the DMA transfer, the DMA performs an **Arbitration Sequence**.
- **Rule 2:** If there is not a higher priority DMA channel pending **DMA Request** and for the ME active DMA channel  $\text{CHCFG.RROAT} = 0_B$  then
  - **Rule 2.1:** The updated TCS shall be written back from the ME to the DMARAM.
  - **Rule 2.2:** During each clock cycle the DMA shall perform an **Arbitration Sequence**.
  - **Rule 2.3:** If a **DMA Request** is received then the ME shall execute a DMA transfer.
  - **Rule 2.4:** On completion of the DMA transfer, the DMA shall perform an **Arbitration Sequence**.
- **Rule 3:** If there is not a higher priority DMA channel pending **DMA Request** and for the ME active DMA channel  $\text{CHCFG.RROAT} = 1_B$  then
  - **Rule 3.1:** The ME shall execute a DMA transfer.
  - **Rule 3.2:** On completion of the DMA transfer, the DMA shall perform an **Arbitration Sequence**.
- **Rule 4:** On completion of the last DMA transfer, the DMA channel request bit is cleared ( $\text{TSR.CH} = 0_B$ ).
- **Rule 5:** The **Arbitration Sequence** continues until all DMA channel access pending requests are serviced by ME.

### 18.3.3.9 DMA Channel Reset

Software shall reset an individual DMA channel by setting the DMA channel reset bit ( $\text{TSR.RST} = 1_B$ ). When a **DMA Channel Reset** is applied to a DMA channel, the transition to the **Reset State** ( $\text{TSR.RST} = 0_B$ ) is as follows:

- **Idle State** and **Pending State**: the DMA channel shall transition to the **Reset State**.
- **Active State**: on completion of the current DMA transfer, the DMA channel transitions to the **Reset State**.

#### Reset State

On completion of a **DMA Channel Reset** the DMA channel enters the **Reset State** defined as:

- The following DMA channel bits are reset:
  - DMA Transaction State Register:  $\text{TSR.HLTREQ}$ ,  $\text{TSR.HLTACK}$ ,  $\text{TSR.HTRE}$ ,  $\text{TSR.CH}$  and  $\text{TSR.TRL}$ .
  - DMARAM TCS:  $\text{CHCFG.PRSEL}$ ,  $\text{CHCSR.ICH}$ ,  $\text{CHCSR.IPM}$ ,  $\text{CHCSR.WRAPD}$ ,  $\text{CHCSR.WRAPPS}$ ,  $\text{CHCSR.FROZEN}$ ,  $\text{CHCSR.BUFFER}$ ,  $\text{CHCSR.LXO}$  and  $\text{CHCHSR.TCOUNT}$ .
- If a circular buffer ADICR.SCBE and/or ADICR.DCBE is enabled for the DMA channel then the source and/or destination address register will be set to the wrap boundary else the address registers are cleared.
- DMA channel shadow address register (SHADR) shall be cleared.

## Direct Memory Access (DMA)

### Resetting a DMA Channel

A user program must execute the following steps to reset a DMA channel:

1. If a **DMA Hardware Request** is enabled then disable hardware requests (TSR.DCH =  $1_B$ ).
2. Software requests a DMA channel reset (TSR.RST =  $1_B$ ).
3. Software shall monitor the DMA channel reset and the DMA channel SADR, DADR and SHADR registers.
4. As soon as the DMA has cleared the DMA channel reset (TSR.RST =  $0_B$ ) and the DMA has reset the DMA channel SADR, DADR and SHADR registers, the **DMA Channel Reset** has completed.

During a **DMA Channel Reset** operation, a **DMA Software Request** must not be initiated by setting CHCSR.SCH =  $1_B$ .

### Restarting a DMA Channel

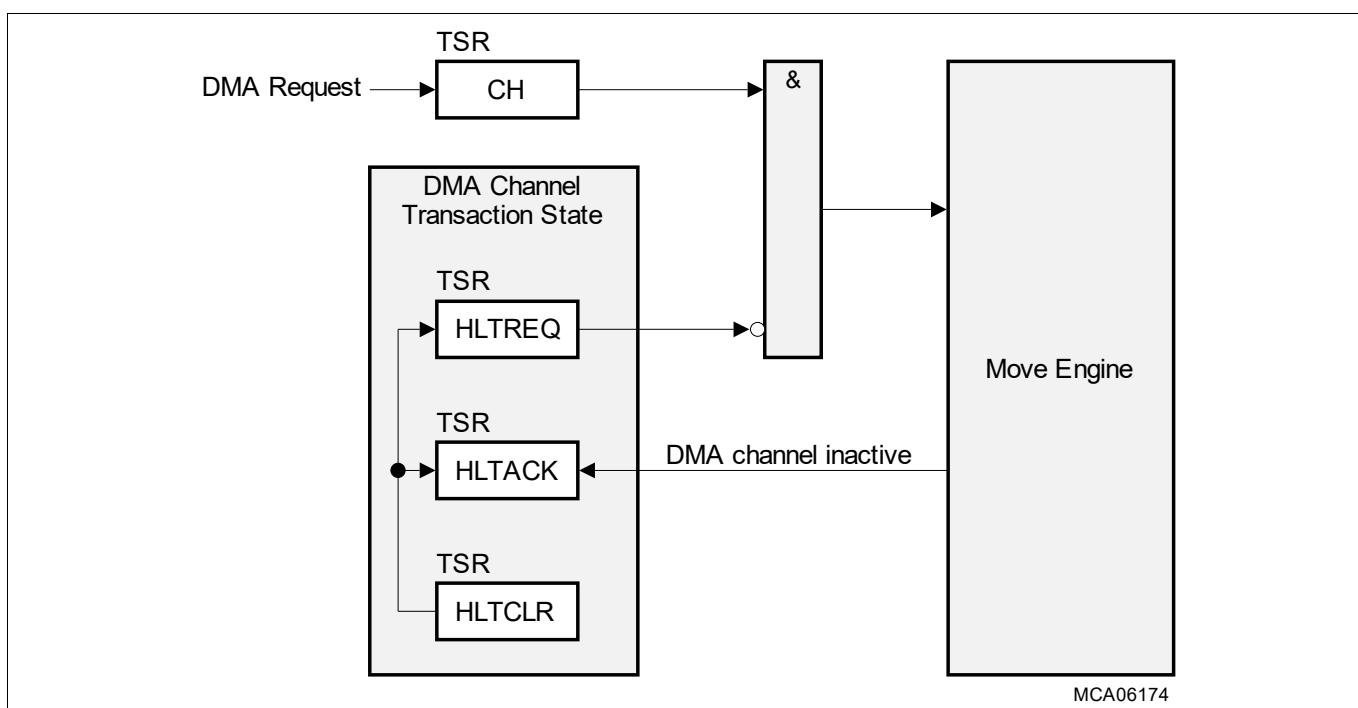
A user program must execute the following steps to restart a DMA channel after a **DMA Channel Reset**:

1. Configure the DMA channel TCS including:
  - a) DMA double buffering: program address pointers to the base address.
2. The DMA channel shall be restarted as follows:
  - a) **DMA Software Request**: software initiates a DMA request (DMA channel CHCSR.SCH =  $1_B$ ).
  - b) **DMA Hardware Request**: software enables a DMA request (DMA channel TSR.ECH =  $1_B$ ).

### 18.3.3.10 DMA Channel Halt

A DMA channel may be halted during a DMA transaction and the state frozen to allow a background RAM test to be run over a destination memory in order to detect stuck bits and distinguish between static errors and transient errors. On completion of the RAM test the DMA channel may be re-started and the DMA transaction completed.

The DMA channel halt logic (at **Figure 195**) utilizes a set/clear mechanism to request the DMA channel transitions to and from the HALT state on completion of the current DMA transfer. Only writing a logic ‘1’ to set or clear a halt request (at **Figure 196**) to a DMA channel has an effect. The status of other DMA channels shall be ignored.



**Figure 195 DMA Channel Halt Logic**

## Direct Memory Access (DMA)

### Halt State

A DMA channel is defined to be in the **Halt State** when DMA channel TSR.HLTACK = 1<sub>B</sub>.

### Entering DMA Channel Halt

A DMA channel shall be halted by software writing the DMA channel halt request bit (TSR.HLTREQ = 1<sub>B</sub>). The DMA channel enters the **Halt State** as follows:

- **Idle State, Reset State** and **Pending State**: as soon as the DMA channel receives the halt request, the DMA channel shall transition to the **Halt State**.
- **Active State**: on completion of the current DMA transfer, the DMA channel shall transition to the **Halt State**.

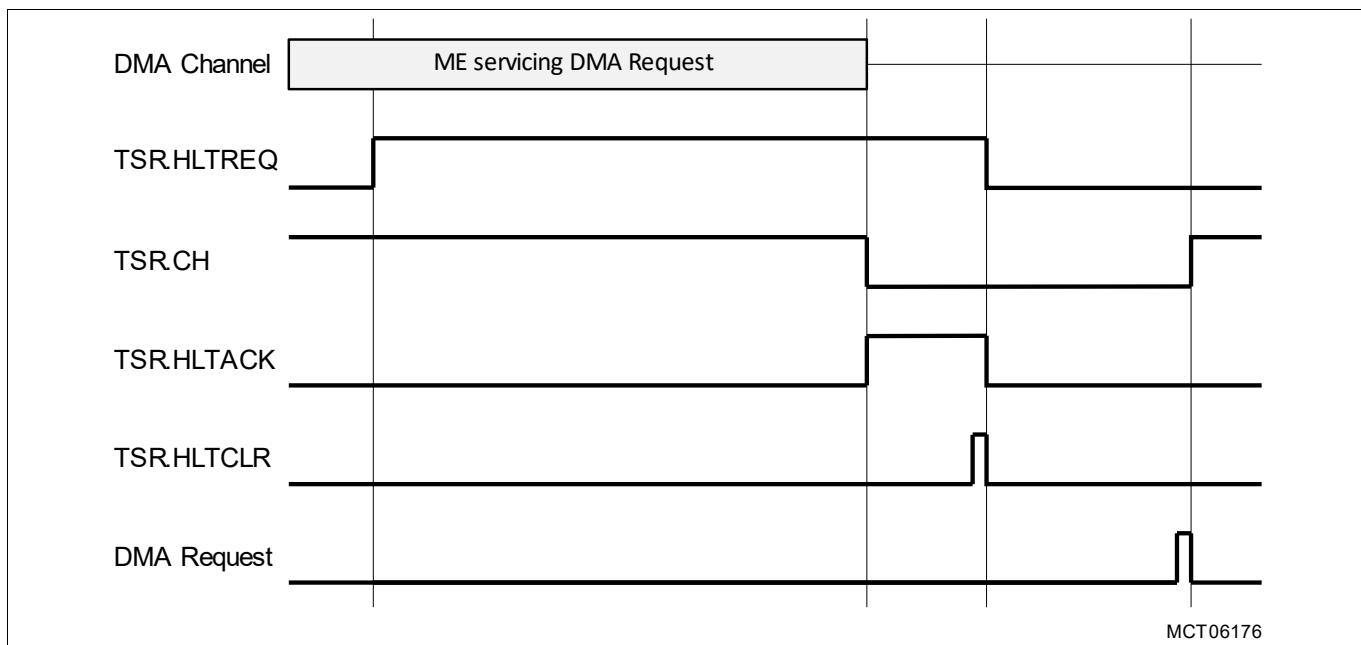


Figure 196 DMA Channel Halt Operation

### Exiting DMA Channel Halt

A DMA channel is released from the **Halt State** by software writing the DMA channel halt clear bit (TSR.HLTCLR = 1<sub>B</sub>). The DMA operation is resumed. If the halt request is cleared before it is acknowledged then there is no effect on DMA operation.

### DMA Channel Hardware Request during DMA Channel Halt

If a DMA channel is in the **Halt State** and Hardware Transaction Request is enabled (TSR.HTRE = 1<sub>B</sub>) then the DMA channel will respond to a **DMA Hardware Request** as follows:

- No DMA request pending (TSR.CH = 0<sub>B</sub>): the DMA channel shall set the access pending bit (TSR.CH = 1<sub>B</sub>). The **DMA Hardware Request** shall be serviced when DMA channel exits the **Halt State**.
- DMA request pending (TSR.CH = 1<sub>B</sub>): the DMA channel shall record a TRL event. If the DMA channel TRL enable bit is set (TSR.ETRL = 1<sub>B</sub>), then the DMA shall trigger a **DMA RP Error Interrupt Service Request**.

## Direct Memory Access (DMA)

## 18.3.4 DMA Random Access Memory

Software stores a DMA channel TCS (at [Figure 197](#)) in the DMARAM to define the move function of each DMA channel.

	WORD 7	WORD 6	WORD 5	WORD 4	WORD 3	WORD 2	WORD 1	WORD 0
CHCSR	SHADR	CHCFGGR	ADICR	DADR	SADR	SDCRCR	RDCRCR	
<b>SCH</b>	SHADR[31]	Reserved	IRDV[3]	DADR[31]	SADR[31]	SDCRC[31]	RDCRC[31]	
Reserved	SHADR[30]	Reserved	IRDV[2]	DADR[30]	SADR[30]	SDCRC[30]	RDCRC[30]	
Reserved	SHADR[29]	Reserved	IRDV[1]	DADR[29]	SADR[29]	SDCRC[29]	RDCRC[29]	
Reserved	SHADR[28]	PRSEL	IRDV[0]	DADR[28]	SADR[28]	SDCRC[28]	RDCRC[28]	
<b>SIT</b>	SHADR[27]	SWAP	INTCT[1]	DADR[27]	SADR[27]	SDCRC[27]	RDCRC[27]	
<b>CICH</b>	SHADR[26]	PATSEL[2]	INTCT[0]	DADR[26]	SADR[26]	SDCRC[26]	RDCRC[26]	
<b>CWRP</b>	SHADR[25]	PATSEL[1]	WRPDE	DADR[25]	SADR[25]	SDCRC[25]	RDCRC[25]	
<b>SWB</b>	SHADR[24]	PATSEL[0]	WRPSE	DADR[24]	SADR[24]	SDCRC[28]	RDCRC[24]	
<b>FROZEN</b>	SHADR[23]	CHDWI[2]	Reserved	DADR[23]	SADR[23]	SDCRC[27]	RDCRC[23]	
<b>BUFFER</b>	SHADR[22]	CHDWI[1]	STAMP	DADR[22]	SADR[22]	SDCRC[26]	RDCRC[22]	
Reserved	SHADR[21]	CHDWI[0]	DCBE	DADR[21]	SADR[21]	SDCRC[21]	RDCRC[21]	
Reserved	SHADR[20]	CHMODE	SCBE	DADR[20]	SADR[20]	SDCRC[20]	RDCRC[20]	
<b>IPM</b>	SHADR[19]	RROAT	SHCT[3]	DADR[19]	SADR[19]	SDCRC[19]	RDCRC[19]	
<b>ICH</b>	SHADR[18]	BLKM[2]	SHCT[2]	DADR[18]	SADR[18]	SDCRC[18]	RDCRC[18]	
<b>WRPD</b>	SHADR[17]	BLKM[1]	SHCT[1]	DADR[17]	SADR[17]	SDCRC[17]	RDCRC[17]	
<b>WRPS</b>	SHADR[16]	BLKM[0]	SHCT[0]	DADR[16]	SADR[16]	SDCRC[16]	RDCRC[16]	
<b>LXO</b>	SHADR[15]	Reserved	CLBD[3]	DADR[15]	SADR[15]	SDCRC[15]	RDCRC[15]	
Reserved	SHADR[14]	Reserved	CLBD[2]	DADR[14]	SADR[14]	SDCRC[14]	RDCRC[14]	
<b>TCOUNT[13]</b>	SHADR[13]	TREL[13]	CLBD[1]	DADR[13]	SADR[13]	SDCRC[13]	RDCRC[13]	
<b>TCOUNT[12]</b>	SHADR[12]	TREL[12]	CLBD[0]	DADR[12]	SADR[12]	SDCRC[12]	RDCRC[12]	
<b>TCOUNT[11]</b>	SHADR[11]	TREL[11]	CLBS[3]	DADR[11]	SADR[11]	SDCRC[11]	RDCRC[11]	
<b>TCOUNT[10]</b>	SHADR[10]	TREL[10]	CLBS[2]	DADR[10]	SADR[10]	SDCRC[10]	RDCRC[10]	
<b>TCOUNT[9]</b>	SHADR[9]	TREL[9]	CLBS[1]	DADR[9]	SADR[9]	SDCRC[9]	RDCRC[9]	
<b>TCOUNT[8]</b>	SHADR[8]	TREL[8]	CLBS[0]	DADR[8]	SADR[8]	SDCRC[8]	RDCRC[8]	
<b>TCOUNT[7]</b>	SHADR[7]	TREL[7]	INCD	DADR[7]	SADR[7]	SDCRC[7]	RDCRC[7]	
<b>TCOUNT[6]</b>	SHADR[6]	TREL[6]	DMF[2]	DADR[6]	SADR[6]	SDCRC[6]	RDCRC[6]	
<b>TCOUNT[5]</b>	SHADR[5]	TREL[5]	DMF[1]	DADR[5]	SADR[5]	SDCRC[5]	RDCRC[5]	
<b>TCOUNT[4]</b>	SHADR[4]	TREL[4]	DMF[0]	DADR[4]	SADR[4]	SDCRC[4]	RDCRC[4]	
<b>TCOUNT[3]</b>	SHADR[3]	TREL[3]	INCS	DADR[3]	SADR[3]	SDCRC[3]	RDCRC[3]	
<b>TCOUNT[2]</b>	SHADR[2]	TREL[2]	SMF[2]	DADR[2]	SADR[2]	SDCRC[2]	RDCRC[2]	
<b>TCOUNT[1]</b>	SHADR[1]	TREL[1]	SMF[1]	DADR[1]	SADR[1]	SDCRC[1]	RDCRC[1]	
<b>TCOUNT[0]</b>	SHADR[0]	TREL[0]	SMF[0]	DADR[0]	SADR[0]	SDCRC[0]	RDCRC[0]	

Key

Read/Write

Read Only

Write Only

MCA06180

Figure 197 DMARAM TCS Organization

## Direct Memory Access (DMA)

### 18.3.4.1 DMA Channel Operation

The DMARAM TCS defines the type of DMA channel operation and the supported on chip bus access size.

**Table 572 DMA Channel Operation**

PATSEL	STAMP	SHCT	Description	BTR4	BTR2	SDTD	SDTW	SDTH	SDTB
000 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	0 <sub>B</sub>	0001 <sub>B</sub>	<b>Shadow Operation Read Only Mode Source Address</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	0 <sub>B</sub>	0010 <sub>B</sub>	<b>Shadow Operation Read Only Mode Destination Address</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	0 <sub>B</sub>	0101 <sub>B</sub>	<b>Shadow Operation Direct Write Mode Source Address</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	0 <sub>B</sub>	0110 <sub>B</sub>	<b>Shadow Operation Direct Write Mode Destination Address</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	0 <sub>B</sub>	1000 <sub>B</sub>	<b>DMA Double Source Buffering Software Switch Only</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	0 <sub>B</sub>	1001 <sub>B</sub>	<b>DMA Double Source Buffering Software Switch and Automatic Hardware Switch</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	0 <sub>B</sub>	1010 <sub>B</sub>	<b>DMA Double Destination Buffering Software Switch Only</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	0 <sub>B</sub>	1011 <sub>B</sub>	<b>DMA Double Destination Buffering Software Switch and Automatic Hardware Switch</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	0 <sub>B</sub>	1100 <sub>B</sub>	<b>DMA Linked List (DMALL)</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	0 <sub>B</sub>	1101 <sub>B</sub>	<b>Accumulated Linked List (ACLLL)</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	0 <sub>B</sub>	1110 <sub>B</sub>	<b>Safe Linked List (SAFLL)</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	0 <sub>B</sub>	1111 <sub>B</sub>	<b>Conditional Linked List (CONLL)</b> DMA read move data compared to PRR0	No	No	No	No	No	Yes
100 <sub>B</sub>	0 <sub>B</sub>	1111 <sub>B</sub>	<b>Conditional Linked List (CONLL)</b> DMA read move data compared to PRR1	No	No	No	No	No	Yes
000 <sub>B</sub>	1 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation with DMA Timestamp</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	1 <sub>B</sub>	0001 <sub>B</sub>	<b>Shadow Operation Read Only Mode Source Address with DMA Timestamp</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	1 <sub>B</sub>	0010 <sub>B</sub>	<b>Shadow Operation Read Only Mode Destination Address with DMA Timestamp</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	1 <sub>B</sub>	0101 <sub>B</sub>	<b>Shadow Operation Direct Write Mode Source Address with DMA Timestamp</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	1 <sub>B</sub>	0110 <sub>B</sub>	<b>Shadow Operation Direct Write Mode Destination Address with DMA Timestamp</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	1 <sub>B</sub>	1001 <sub>B</sub>	<b>DMA Double Source Buffering with DMA Timestamp<sup>1)</sup></b> Software Switch and Automatic Hardware Switch	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	1 <sub>B</sub>	1011 <sub>B</sub>	<b>DMA Double Destination Buffering with DMA Timestamp<sup>1)</sup></b> Software Switch and Automatic Hardware Switch	Yes	Yes	Yes	Yes	Yes	Yes

## Direct Memory Access (DMA)

Table 572 DMA Channel Operation (cont'd)

PATSEL	STAMP	SHCT	Description	BTR4	BTR2	SDTD	SDTW	SDTH	SDTB
000 <sub>B</sub>	1 <sub>B</sub>	1100 <sub>B</sub>	<b>DMA Linked List (DMALL) with DMA Timestamp</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	1 <sub>B</sub>	1101 <sub>B</sub>	<b>Accumulated Linked List (ACLL) with DMA Timestamp</b>	Yes	Yes	Yes	Yes	Yes	Yes
000 <sub>B</sub>	1 <sub>B</sub>	1110 <sub>B</sub>	<b>Safe Linked List (SAFLL) with DMA Timestamp</b>	Yes	Yes	Yes	Yes	Yes	Yes
001 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 8-bit Channel Data Width</b> <b>Lower Pattern Compare to PRR0</b>	No	No	No	No	No	Yes
010 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 8-bit Channel Data Width</b> <b>Upper Pattern Compare to PRR0</b>	No	No	No	No	No	Yes
011 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 8-bit Channel Data Width</b> <b>Two Byte Pattern Detection Sequence compared to PRR0</b>	No	No	No	No	No	Yes
001 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 16-bit Channel Data Width</b> <b>Aligned Mode compared to PRR0</b>	No	No	No	No	Yes	No
010 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 16-bit Channel Data Width</b> <b>Unaligned Mode 1 or Unaligned Mode 2 compared to PRR0</b>	No	No	No	No	Yes	No
011 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 16-bit Channel Data Width</b> <b>Combined Mode compared to PRR0</b>	No	No	No	No	Yes	No
001 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 32-bit Channel Data Width</b> <b>Lower Data Half Word compared to PRR0</b>	No	No	No	Yes	No	No
010 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 32-bit Channel Data Width</b> <b>Upper Data Half Word compared to PRR0</b>	No	No	No	Yes	No	No
011 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 32-bit Channel Data Width</b> <b>Complete Data Word compared to PRR0</b>	No	No	No	Yes	No	No
101 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 8-bit Channel Data Width</b> <b>Lower Pattern Compare to PRR1</b>	No	No	No	No	No	Yes
110 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 8-bit Channel Data Width</b> <b>Upper Pattern Compare to PRR1</b>	No	No	No	No	No	Yes
111 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 8-bit Channel Data Width</b> <b>Two Byte Pattern Detection Sequence compared to PRR1</b>	No	No	No	No	No	Yes
101 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 16-bit Channel Data Width</b> <b>Aligned Mode compared to PRR1</b>	No	No	No	No	Yes	No

## Direct Memory Access (DMA)

Table 572 DMA Channel Operation (cont'd)

PATSEL	STAMP	SHCT	Description	BTR4	BTR2	SDTD	SDTW	SDTH	SDTB
110 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 16-bit Channel Data Width</b> <b>Unaligned Mode 1 or Unaligned Mode 2 compared to PRR1</b>	No	No	No	No	Yes	No
111 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 16-bit Channel Data Width</b> <b>Combined Mode compared to PRR1</b>	No	No	No	No	Yes	No
101 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 32-bit Channel Data Width</b> <b>Lower Data Half Word compared to PRR1</b>	No	No	No	Yes	No	No
110 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 32-bit Channel Data Width</b> <b>Upper Data Half Word compared to PRR1</b>	No	No	No	Yes	No	No
111 <sub>B</sub>	0 <sub>B</sub>	0000 <sub>B</sub>	<b>Move Operation<sup>2)</sup></b> <b>Pattern Detection for 32-bit Channel Data Width</b> <b>Complete Data Word compared to PRR1</b>	No	No	No	Yes	No	No
Others		<b>Reserved<sup>3)</sup></b>		N/A	N/A	N/A	N/A	N/A	N/A

- 1) DMA timestamp shall only be appended on completion of a DMA transaction.
- 2) DMA channel operation “Pattern Detection” shall function for all types of DMA channel Shadow Operation.
- 3) If a DMA channel is configured for a reserved value of DMA channel ADICR.SHCT and a DMA request is received, the DMA clears DMA channel TSR.CH and performs no DMA moves.

## 18.3.4.2 DMA Channel Updates

Software shall only configure the DMARAM channel TCS when the DMA channel is in the **Idle State** (DMA channel CHCSR.TCOUNT=0<sub>D</sub>) or **Reset State**. If the DMA channel is in the **Idle State** (DMA channel CHCSR.TCOUNT!=0<sub>D</sub>), **Pending State** or **Active State**, software shall not update the DMARAM channel TCS. The following DMA channel update exceptions apply for **Shadow Operations** and **Double Buffering Operations**:

## 18.3.4.2.1 Shadow Operations

If the DMARAM channel TCS is configured for **Shadow Operation** and the DMA channel is in the **Idle State** (DMA channel CHCSR.TCOUNT!=0<sub>D</sub>), **Pending State** or **Active State**, software shall be restricted to updating the DMARAM channel TCS words marked ‘X’:

Table 573 DMARAM channel TCS updates during Shadow Operation

SHCT	Description	CHCSR	SHADR	CHCFGR	ADICR	DADR	SADR	SDCRCR	RDCRCR
0001 <sub>B</sub>	<b>Shadow Operation Read Only Mode</b> <b>Source Address</b>						X		
0010 <sub>B</sub>	<b>Shadow Operation Read Only Mode</b> <b>Destination Address</b>					X			

## Direct Memory Access (DMA)

**Table 573 DMARAM channel TCS updates during Shadow Operation (cont'd)**

SHCT	Description	CHCSR	SHADR	CHCFG	ADICR	DADR	SADR	SDCRCR	RDCRCR
0101 <sub>B</sub>	<b>Shadow Operation Direct Write Mode</b> Source Address		X						
0110 <sub>B</sub>	<b>Shadow Operation Direct Write Mode</b> Destination Address		X						

### 18.3.4.2.2 Double Buffering Operations

If the DMARAM channel TCS is configured for **Double Buffering Operations** and the DMA channel is in the **Idle State** (DMA channel CHCSR.TCOUNT!=0<sub>D</sub>), **Pending State** or **Active State**, software shall be restricted to updating the DMARAM channel TCS words marked 'X':

**Table 574 DMARAM channel updates during Double Buffering Operations**

SHCT	Description	CHCSR	SHADR	CHCFG	ADICR	DADR	SADR	SDCRCR	RDCRCR
1000 <sub>B</sub>	<b>DMA Double Source Buffering</b> <b>Software Switch Only</b> DMA channel CHCSR.BUFFER = 0 <sub>B</sub>	X	X						
1000 <sub>B</sub>	<b>DMA Double Source Buffering</b> <b>Software Switch Only</b> DMA channel CHCSR.BUFFER = 1 <sub>B</sub>	X					X		
1001 <sub>B</sub>	<b>DMA Double Source Buffering</b> <b>Software Switch and Automatic Hardware Switch</b> DMA channel CHCSR.BUFFER = 0 <sub>B</sub>	X	X						
1001 <sub>B</sub>	<b>DMA Double Source Buffering</b> <b>Software Switch and Automatic Hardware Switch</b> DMA channel CHCSR.BUFFER = 1 <sub>B</sub>	X					X		
1010 <sub>B</sub>	<b>DMA Double Destination Buffering</b> <b>Software Switch Only</b> DMA channel CHCSR.BUFFER = 0 <sub>B</sub>	X	X						
1010 <sub>B</sub>	<b>DMA Double Destination Buffering</b> <b>Software Switch Only</b> DMA channel CHCSR.BUFFER = 1 <sub>B</sub>	X				X			
1011 <sub>B</sub>	<b>DMA Double Destination Buffering</b> <b>Software Switch and Automatic Hardware Switch</b> DMA channel CHCSR.BUFFER = 0 <sub>B</sub>	X	X						
1011 <sub>B</sub>	<b>DMA Double Destination Buffering</b> <b>Software Switch and Automatic Hardware Switch</b> DMA channel CHCSR.BUFFER = 1 <sub>B</sub>	X				X			

### 18.3.4.3 DMA Channel Reconfiguration

If a DMA channel is to be configured, software shall apply a **DMA Channel Reset** to initialize the DMA channel.

## Direct Memory Access (DMA)

After the DMA channel has entered **Reset State**, software shall configure the DMA channel for a DMA operation.

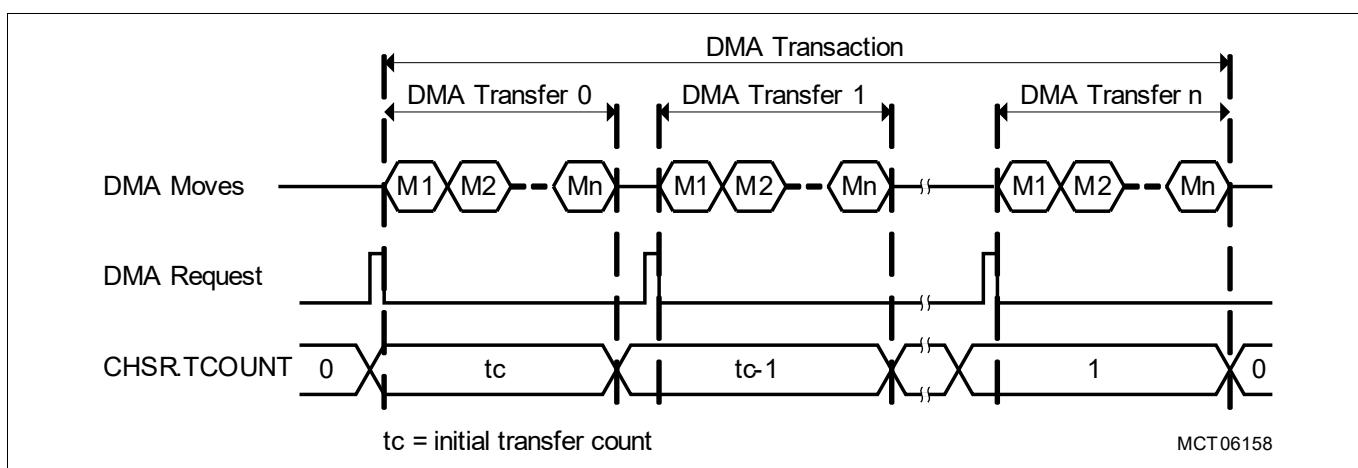
### 18.3.4.4 Move Operation

The number of DMA moves (at **Figure 198**) may be calculated from the following:

- Block Mode (CHCFGR.BLKM) defines the number of DMA moves in a DMA transfer.
- Transfer Reload (CHCFGR.TREL) defines the number of DMA transfers in a DMA transaction.

After a DMA move, the next source and destination addresses are calculated. Source and destination addresses are calculated independently of each other. The following address calculation parameters may be selected:

- The address offset, which is a multiple of the selected data width.
- The offset direction: addition, subtraction, or none (unchanged address).



**Figure 198 Block Mode and Transfer Count**

#### 18.3.4.4.1 Address Generation

Address control bits (in DMA channel ADICR) determine how the addresses increment or decrement. The data width is defined in CHCFGR.CHDW and is taken into account during the address calculation. The Address Offset Calculation Tables (at **Table 575** and **Table 576**) show the offset values that are added or subtracted to/from the source address (SMF and INCS parameters) and destination address (DMF and INCD parameters) after each DMA move.

**Table 575 Address Offset Calculation Table (8-bit, 16-bit and 32-bit)**

CHCFGR.CHDW = 000 <sub>B</sub> (8-bit Data Width)			CHCFGR.CHDW = 001 <sub>B</sub> (16-bit Data Width)			CHCFGR.CHDW = 010 <sub>B</sub> (32-bit Data Width)		
SMF DMF	INCS INCD	Address Offset	SMF DMF	INCS INCD	Address Offset	SMF DMF	INCS INCD	Address Offset
000 <sub>B</sub>	0 <sub>B</sub>	-1	000 <sub>B</sub>	0 <sub>B</sub>	-2	000 <sub>B</sub>	0 <sub>B</sub>	-4
	1 <sub>B</sub>	+1		1 <sub>B</sub>	+2		1 <sub>B</sub>	+4
001 <sub>B</sub>	0 <sub>B</sub>	-2	001 <sub>B</sub>	0 <sub>B</sub>	-4	001 <sub>B</sub>	0 <sub>B</sub>	-8
	1 <sub>B</sub>	+2		1 <sub>B</sub>	+4		1 <sub>B</sub>	+8
010 <sub>B</sub>	0 <sub>B</sub>	-4	010 <sub>B</sub>	0 <sub>B</sub>	-8	010 <sub>B</sub>	0 <sub>B</sub>	-16
	1 <sub>B</sub>	+4		1 <sub>B</sub>	+8		1 <sub>B</sub>	+16

## Direct Memory Access (DMA)

Table 575 Address Offset Calculation Table (8-bit, 16-bit and 32-bit) (cont'd)

CHCFGR.CHDW = 000 <sub>B</sub> (8-bit Data Width)			CHCFGR.CHDW = 001 <sub>B</sub> (16-bit Data Width)			CHCFGR.CHDW = 010 <sub>B</sub> (32-bit Data Width)		
SMF DMF	INCS INCD	Address Offset	SMF DMF	INCS INCD	Address Offset	SMF DMF	INCS INCD	Address Offset
011 <sub>B</sub>	0 <sub>B</sub>	-8	011 <sub>B</sub>	0 <sub>B</sub>	-16	011 <sub>B</sub>	0 <sub>B</sub>	-32
	1 <sub>B</sub>	+8		1 <sub>B</sub>	+16		1 <sub>B</sub>	+32
100 <sub>B</sub>	0 <sub>B</sub>	-16	100 <sub>B</sub>	0 <sub>B</sub>	-32	100 <sub>B</sub>	0 <sub>B</sub>	-64
	1 <sub>B</sub>	+16		1 <sub>B</sub>	+32		1 <sub>B</sub>	+64
101 <sub>B</sub>	0 <sub>B</sub>	-32	101 <sub>B</sub>	0 <sub>B</sub>	-64	101 <sub>B</sub>	0 <sub>B</sub>	-128
	1 <sub>B</sub>	+32		1 <sub>B</sub>	+64		1 <sub>B</sub>	+128
110 <sub>B</sub>	0 <sub>B</sub>	-64	110 <sub>B</sub>	0 <sub>B</sub>	-128	110 <sub>B</sub>	0 <sub>B</sub>	-256
	1 <sub>B</sub>	+64		1 <sub>B</sub>	+128		1 <sub>B</sub>	+256
111 <sub>B</sub>	0 <sub>B</sub>	-128	111 <sub>B</sub>	0 <sub>B</sub>	-256	111 <sub>B</sub>	0 <sub>B</sub>	-512
	1 <sub>B</sub>	+128		1 <sub>B</sub>	+256		1 <sub>B</sub>	+512

Table 576 Address Offset Calculation Table (64-bit, 128-bit and 256-bit)

CHCFGR.CHDW = 011 <sub>B</sub> (64-bit Data Width)			CHCFGR.CHDW = 100 <sub>B</sub> (128-bit Data Width)			CHCFGR.CHDW = 101 <sub>B</sub> (256-bit Data Width)		
SMF DMF	INCS INCD	Address Offset	SMF DMF	INCS INCD	Address Offset	SMF DMF	INCS INCD	Address Offset
000 <sub>B</sub>	0 <sub>B</sub>	-8	000 <sub>B</sub>	0 <sub>B</sub>	-16	000 <sub>B</sub>	0 <sub>B</sub>	-32
	1 <sub>B</sub>	+8		1 <sub>B</sub>	+16		1 <sub>B</sub>	+32
001 <sub>B</sub>	0 <sub>B</sub>	-16	001 <sub>B</sub>	0 <sub>B</sub>	-32	001 <sub>B</sub>	0 <sub>B</sub>	-64
	1 <sub>B</sub>	+16		1 <sub>B</sub>	+32		1 <sub>B</sub>	+64
010 <sub>B</sub>	0 <sub>B</sub>	-32	010 <sub>B</sub>	0 <sub>B</sub>	-64	010 <sub>B</sub>	0 <sub>B</sub>	-128
	1 <sub>B</sub>	+32		1 <sub>B</sub>	+64		1 <sub>B</sub>	+128
011 <sub>B</sub>	0 <sub>B</sub>	-64	011 <sub>B</sub>	0 <sub>B</sub>	-128	011 <sub>B</sub>	0 <sub>B</sub>	-256
	1 <sub>B</sub>	+64		1 <sub>B</sub>	+128		1 <sub>B</sub>	+256
100 <sub>B</sub>	0 <sub>B</sub>	-128	100 <sub>B</sub>	0 <sub>B</sub>	-256	100 <sub>B</sub>	0 <sub>B</sub>	-512
	1 <sub>B</sub>	+128		1 <sub>B</sub>	+256		1 <sub>B</sub>	+512
101 <sub>B</sub>	0 <sub>B</sub>	-256	101 <sub>B</sub>	0 <sub>B</sub>	-512	101 <sub>B</sub>	0 <sub>B</sub>	-1024
	1 <sub>B</sub>	+256		1 <sub>B</sub>	+512		1 <sub>B</sub>	+1024
110 <sub>B</sub>	0 <sub>B</sub>	-512	110 <sub>B</sub>	0 <sub>B</sub>	-1024	110 <sub>B</sub>	0 <sub>B</sub>	-2048
	1 <sub>B</sub>	+512		1 <sub>B</sub>	+1024		1 <sub>B</sub>	+2048
111 <sub>B</sub>	0 <sub>B</sub>	-1024	111 <sub>B</sub>	0 <sub>B</sub>	-2048	111 <sub>B</sub>	0 <sub>B</sub>	-4096
	1 <sub>B</sub>	+1024		1 <sub>B</sub>	+2048		1 <sub>B</sub>	+4096

## 18.3.4.4.2 Address Calculation Examples

The following example demonstrate address generation for a data width of 16-bit (CHCFGR.CHDW = 001<sub>B</sub>):

## Direct Memory Access (DMA)

### Programmable Address Generation - Example 1

16-bit half-words are moved from a source memory with an incrementing source address offset of  $10_{\text{H}}$  to a destination memory with a decrementing destination address offset of  $08_{\text{H}}$ .

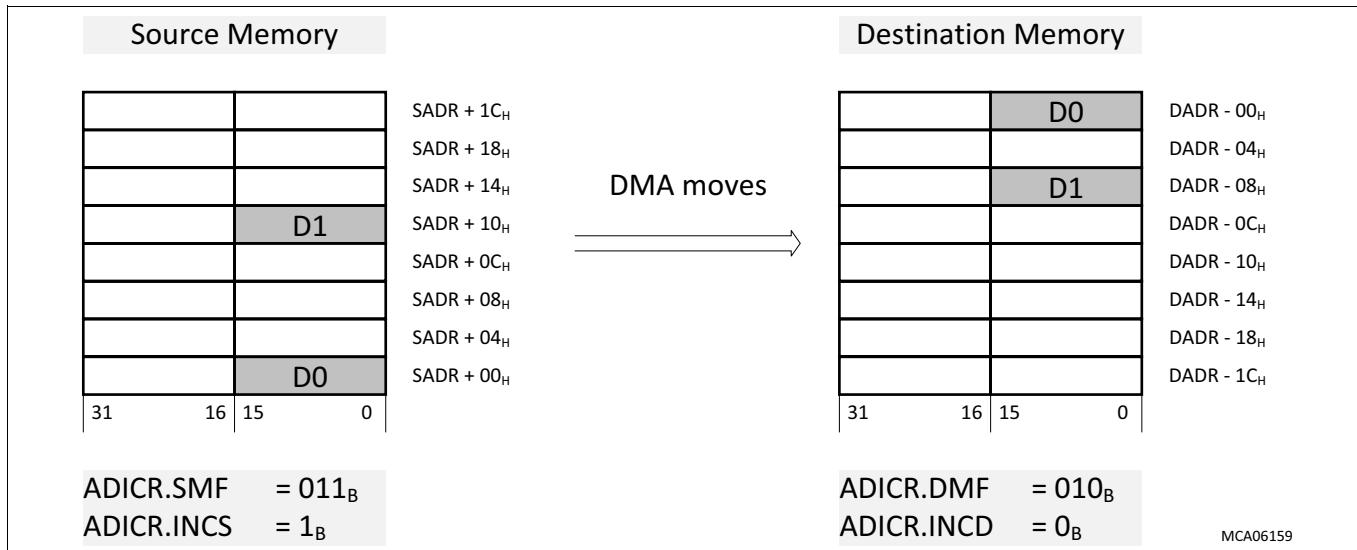


Figure 199 Programmable Address Generation - Example 1

### Programmable Address Generation - Example 2

16-bit half-words are moved from a source memory with an incrementing source address offset of  $02_{\text{H}}$  to a destination memory with an incrementing destination address offset of  $04_{\text{H}}$ .

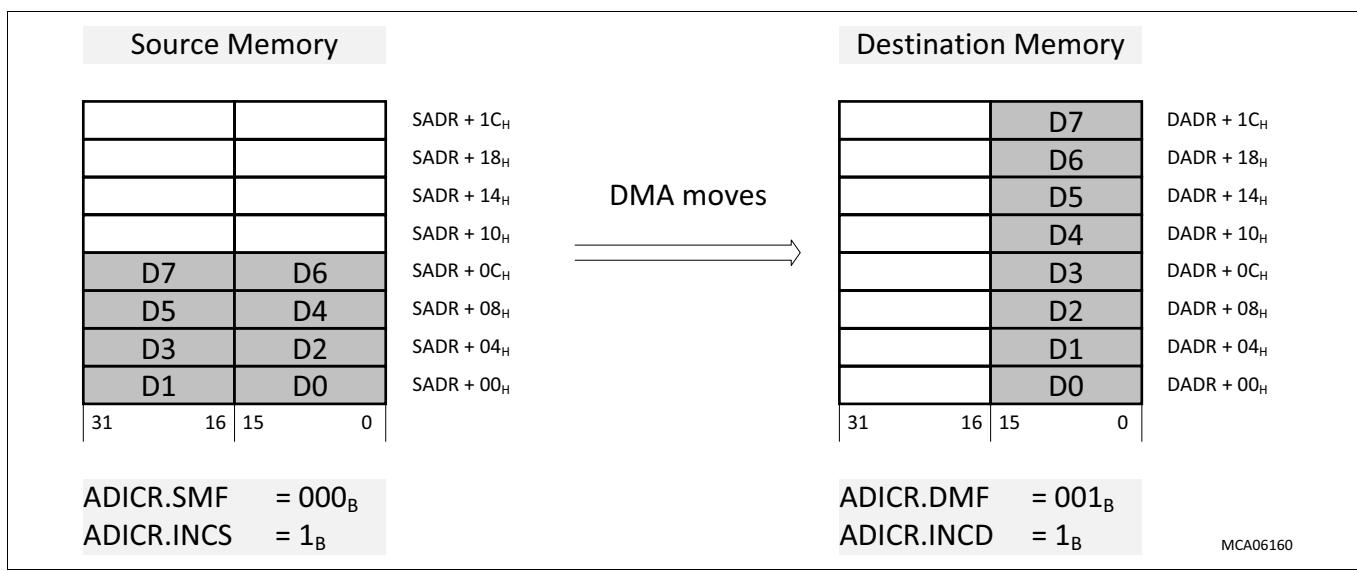


Figure 200 Programmable Address Generation - Example 2

### 18.3.4.4.3 Circular Buffer

The source and destination circular buffers are enabled by setting the DMA channel circular buffer enable bits ADICR.SCBE and ADICR.DCBE respectively. The source address and destination address may be configured to build a circular buffer separately for source and destination data. Within a circular buffer, the addresses are updated within the circular buffer wrap-around limits. The circular buffer length is determined by bit fields ADICR.CBLS (for the source buffer) and ADICR.CBLD (for the destination buffer). These 4-bit wide bit fields determine which bits of the 32-bit address remain unchanged during an address update. Possible buffer sizes of

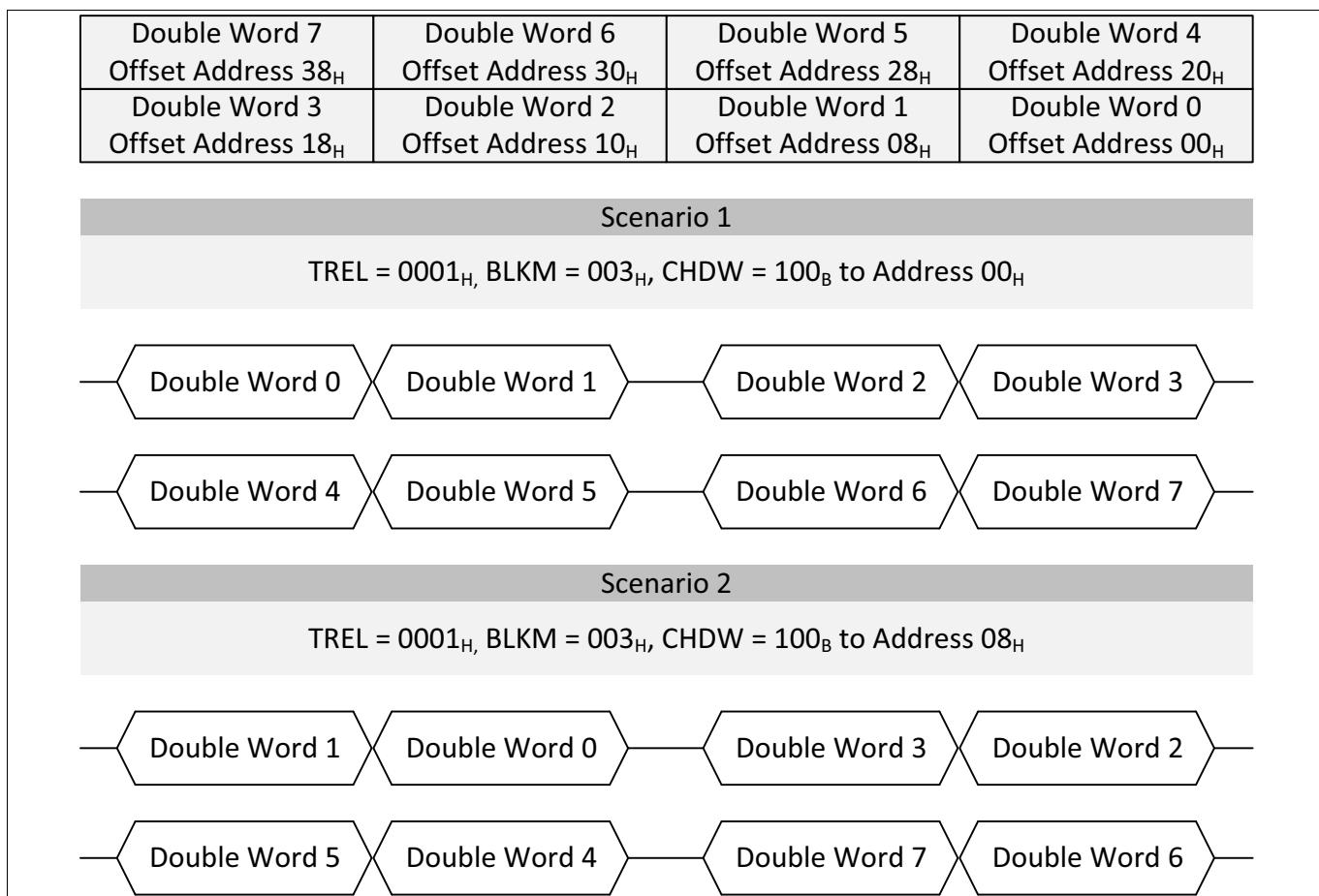
## Direct Memory Access (DMA)

the circular buffers can be  $2^{\text{CBLS}}$  or  $2^{\text{CBLD}}$  bytes (=1, 2, 4, 8, 16, ... up to 64k bytes). Source and destination addresses are incremented or decremented during a DMA move, all upper bits [31:CBLS] of source address and [31:CBLD] of destination address are frozen and remain unchanged, even if a wrap-around from the lower address bits [CBLS-1:0] or [CBLD-1:0] occurred. This address-freezing mechanism always causes the circular buffers to be aligned to a multiple integer value of its size. If the circular buffer size is less than or equal to the selected address offset then the same circular buffer address will always be accessed.

### 18.3.4.4.4 Address Alignment

The DMA shall comply with the SRI bus protocol. All source addresses and destination addresses shall be aligned to the correct address boundary defined by the channel data width (CHCFGR.CHDW) as follows:

- Single transfers: the source address and destination address boundaries shall align with the channel data width (byte, half-word, word or double-word).
- Block transfers: the source address and destination address boundaries shall align to a double-word boundary. The SRI bus protocol defines a wrap around addressing scheme for block transfers not starting with a block transfer sized start address. The effect on a DMA transaction is demonstrated by different scenarios for BTR2 (at [Figure 201](#)) and BTR4 (at [Figure 202](#)).



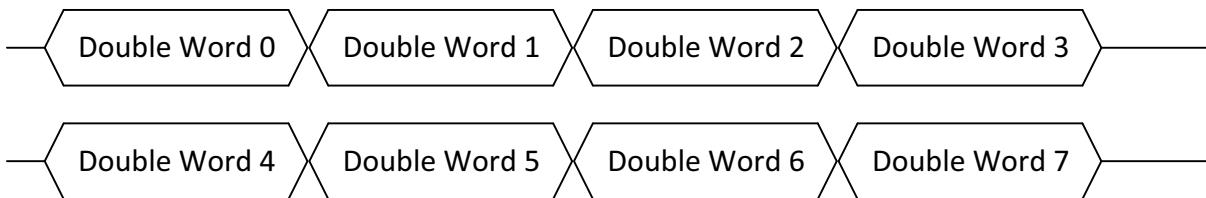
**Figure 201 SRI Bus Protocol Block Transfer Request - 2 Transfers**

## Direct Memory Access (DMA)

Double Word 7 Offset Address 38 <sub>H</sub>	Double Word 6 Offset Address 30 <sub>H</sub>	Double Word 5 Offset Address 28 <sub>H</sub>	Double Word 4 Offset Address 20 <sub>H</sub>
Double Word 3 Offset Address 18 <sub>H</sub>	Double Word 2 Offset Address 10 <sub>H</sub>	Double Word 1 Offset Address 08 <sub>H</sub>	Double Word 0 Offset Address 00 <sub>H</sub>

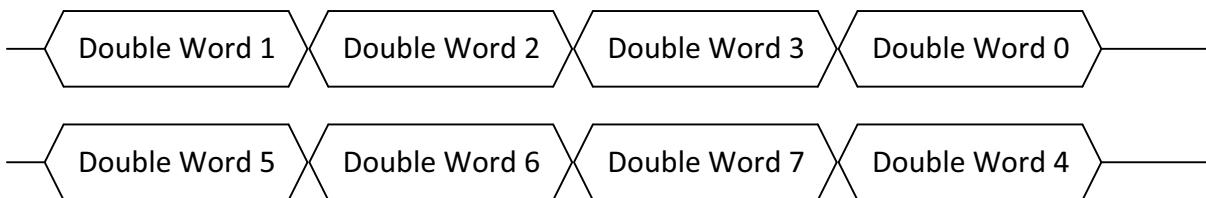
### Scenario 1

TREL = 0001<sub>H</sub>, BLKM = 001<sub>H</sub>, CHDW = 101<sub>B</sub> to Address 00<sub>H</sub>



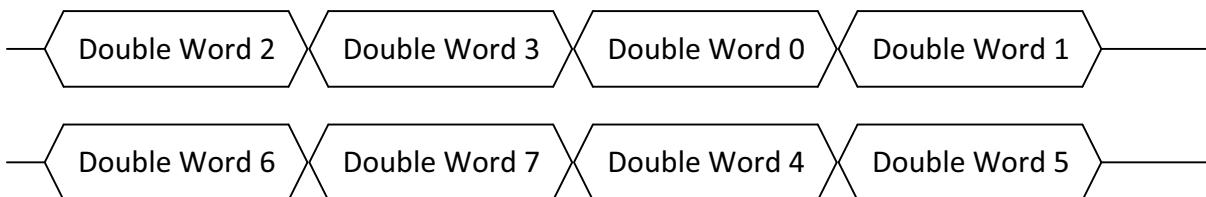
### Scenario 2

TREL = 0001<sub>H</sub>, BLKM = 001<sub>H</sub>, CHDW = 101<sub>B</sub> to Address 08<sub>H</sub>



### Scenario 3

TREL = 0001<sub>H</sub>, BLKM = 001<sub>H</sub>, CHDW = 101<sub>B</sub> to Address 10<sub>H</sub>



### Scenario 4

TREL = 0001<sub>H</sub>, BLKM = 001<sub>H</sub>, CHDW = 101<sub>B</sub> to Address 18<sub>H</sub>

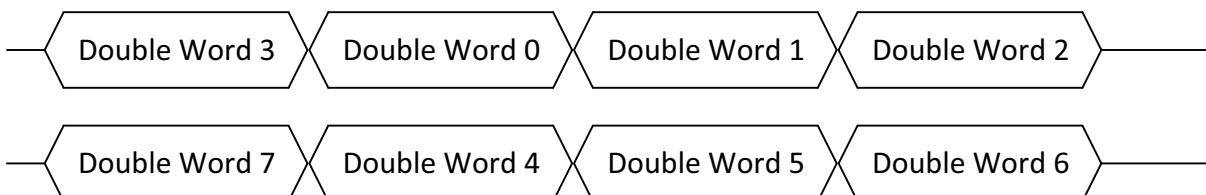


Figure 202 SRI Bus Protocol Block Transfer Request - 4 Transfers

## Direct Memory Access (DMA)

### 18.3.4.4.5 Address Counter

If the source/destination circular buffer is not enabled (ADICR.SCBE/DCBE = 0<sub>B</sub>) then the source/destination address will increment or decrement across the entire 32-bit address field. The address offset is determined by the DMA channel CHCFGR.CHDW. The address will wrap around on the 32-bit address boundary:

- If the address is incrementing then will wrap from FFFFFFFF<sub>H</sub> to 00000000<sub>H</sub>.
- If the address is decrementing then will wrap from 00000000<sub>H</sub> to FFFFFFFF<sub>H</sub>.

### 18.3.4.4.6 DMA Address Checksum

The DMA shall calculate a **SDCRC** checksum in accordance with the IEEE 802.3 standard using the source and destination addresses presented to the on chip bus as input data. Software shall compare the calculated **DMA Address Checksum (SDCRC)** with an expected **DMA Address Checksum** stored in memory to validate the address generation during a DMA transaction. To check the DMA channel address generation use the following steps:

- Software shall initialize the DMA channel **DMA Address Checksum** stored in DAMRAM to a known initial value.
- The DMA channel receives a DMA request.
- If the DMA channel is in the **Active State** and no error or retry event is reported, the ME shall calculate an updated **DMA Address Checksum** for each DMA read move and DMA write move.
- On completion of the DMA transaction, the DMA shall store the final **DMA Address Checksum** in DMARAM.
- Software shall compare the calculated **DMA Address Checksum** and expected **DMA Address Checksum**.
  - If the calculated and expected **DMA Address Checksum match**, then the DMA has correctly calculated the source and destination address. If the checksums do not match, then software shall report a fault.

The **DMA Address Checksum** is not available for the following configurations of **DMA Channel Operation**:

- **DMA Double Source Buffering Software Switch Only**
- **DMA Double Source Buffering Software Switch** and **Automatic Hardware Switch**
- **DMA Double Destination Buffering Software Switch Only**
- **DMA Double Destination Buffering Software Switch** and **Automatic Hardware Switch**
- **Conditional Linked List (CONLL)**

### 18.3.4.4.7 DMA Channel Interrupt Service Request

Each DMA channel generates an interrupt service request (at [Figure 203](#)) to report DMA channel events:

- **DMA Channel Transfer Interrupt Service Request**
- **DMA Channel Pattern Match Interrupt Service Request**
- **DMA Channel Wrap Buffer Interrupt Service Request**

### 18.3.4.4.8 DMA Channel Transfer Interrupt Service Request

If a DMA channel is in the **Active State** then a DMA channel transfer interrupt service request may be activated on completion of a DMA transfer, or when ME CHSR.TCOUNT matches with the value of bit field ADICR.IRDV after it has been decremented after a DMA transfer. An interrupt service request from a DMA channel is indicated when status flag CSR.ICH is set. The status flag shall be reset by software setting DMA channel CHCSR.CICH = 1<sub>B</sub> (or TSR.RST = 1<sub>B</sub>). The DMA channel interrupt service request is enabled when DMA channel ADICR.INTCT[1] is set.

Bit ADICR.INTCT[0] selects one of two types of interrupt source. For the compare operation, bit field ADICR.IRDV (4-bit) is zero-extended to 14-bit and then compared with the 14-bit TCOUNT value. This means that a TCOUNT match interrupt may be generated after one of the last 16 DMA transfers of a DMA transaction. Note that with IRDV = 0000<sub>B</sub>, the match interrupt is generated at the end of a DMA transaction (after the last DMA transfer).

### Direct Memory Access (DMA)

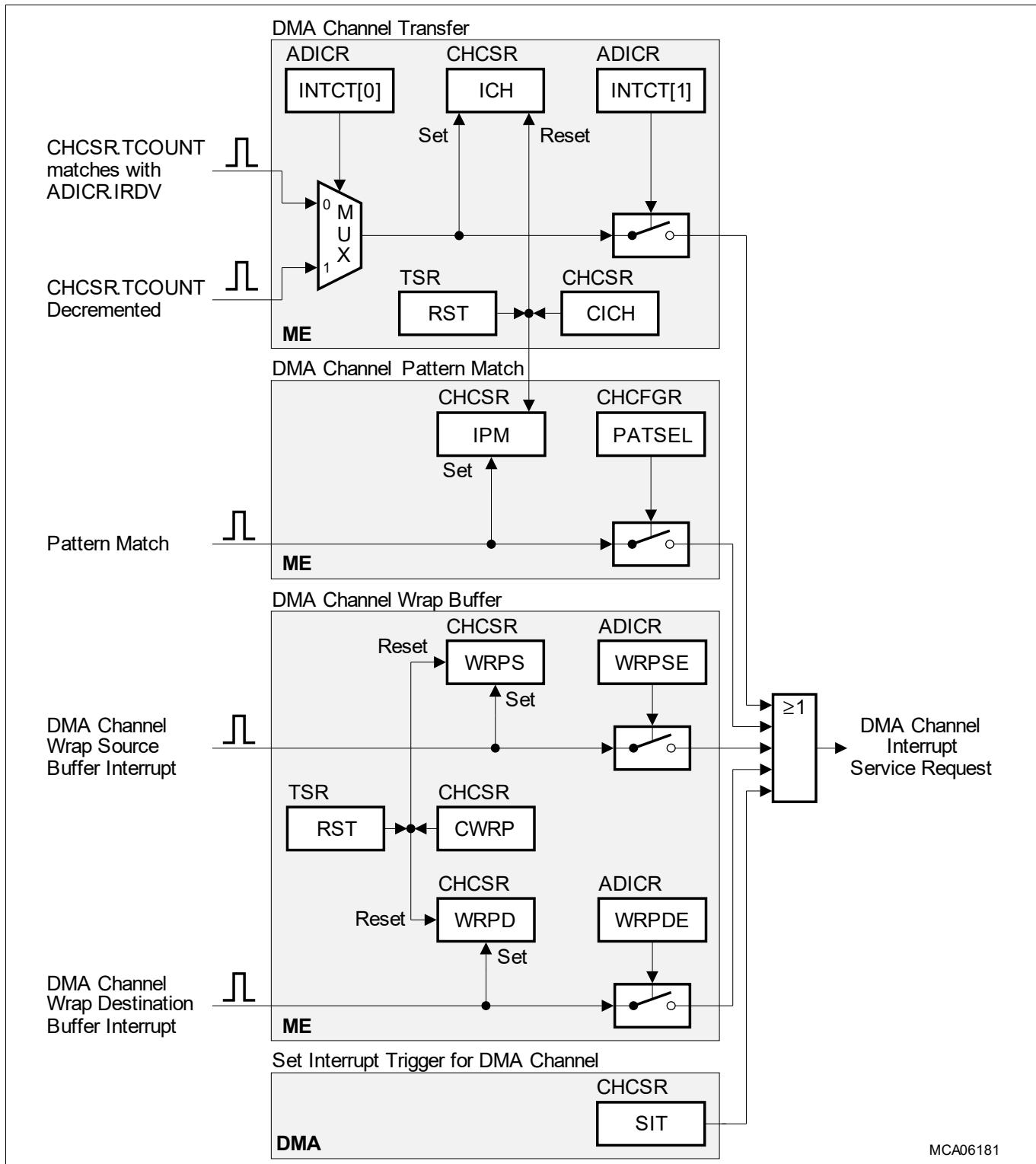


Figure 203 DMA Channel Interrupt Service Request

#### 18.3.4.4.9 DMA Channel Pattern Match Interrupt Service Request

The DMA channel pattern detection interrupt service request is enabled when CHCFGR.PATSEL[1:0] is not equal to  $00_B$ . The pattern detection interrupt is indicated when status flag CHCSR.IPM is set. The status flag CHCSR.IPM shall be reset by software setting bit DMA channel CHCSR.CICH =  $1_B$  (or TSR.RST =  $1_B$ ).

## Direct Memory Access (DMA)

### 18.3.4.4.10 DMA Channel Wrap Buffer Interrupt Service Request

Each DMA channel reports a source buffer wraparound and/or a destination buffer wraparound as follows:

- A wrap source buffer wraparound is indicated by status flag CHCSR.WRPS.
- A wrap destination buffer wraparound is indicated by the status flag CHCSR.WRPD.

Software may reset both buffer wraparound status flags by setting CHCSR.CWRP = 1<sub>B</sub> (or TSR.RST = 1<sub>B</sub>).

A DMA channel wrap buffer interrupt service request is enabled as follows:

- A DMA channel wrap source buffer interrupt service request is enabled when bit ADICR.WRPSE is set.
- A DMA channel wrap destination buffer interrupt service request is enabled when bit ADICR.WRPDE is set.

The DMA channel wrap source buffer wraparound interrupt service request and the DMA channel wrap destination buffer wraparound interrupt service request are OR-ed together with other DMA channel interrupt service requests to form one DMA channel interrupt service request.

### 18.3.4.5 Shadow Operation

If a DMA channel is configured for shadow operation, the DMA buffers a variable source or destination address.

#### 18.3.4.5.1 Application of Shadow Operation

In a typical application, an ASC module that receives data (fixed DMA channel source address) has to deliver the data to a memory buffer using a DMA transaction (variable DMA channel destination address). After a certain amount of data has been transferred, a new DMA transaction should be initiated to deliver further ASC data into another memory buffer. While the destination address register is updated during a running DMA transaction with the actual destination address, a shadow mechanism allows programming of a new destination address without disturbing the content of the destination address register. In this case, the new destination address is written and buffered in the shadow address register. At the start of the next DMA transaction, the destination address register shall use the new address without CPU intervention. The shadow operation avoids the CPU having to check for the end of a DMA transaction before reprogramming address registers.

#### 18.3.4.5.2 Shadowed Address Register

The shadow address register stores either a new source address, or a new destination address. If both the source address and destination address have to be updated for the next DMA transaction, a running DMA transaction for this DMA channel must be finished. After that, source and destination address registers may be written before the next DMA transaction is started.

Only one address register may be shadowed while a DMA transaction is running, because the shadow register can only be assigned either to the source address or to the destination address. Note that the shadow address transfer mechanism has the same behavior in **Single Mode** and **Continuous Mode**.

The function of the shadow address register shall be defined by the mode of the shadow operation:

- **Read Only Mode:** if software writes to the shadow address register, the DMA shall return a bus error.
- **Direct Write Mode:** if software writes to the shadow address register, the DMA shall store a new shadow address in the shadow address register.

#### 18.3.4.5.3 Read Only Mode

If software writes a new DMA channel source or destination address value, the DMA shall store the new address value as follows:

- **Idle State** (DMA channel CHCSR.TCOUNT=0)

## Direct Memory Access (DMA)

- If software writes a new DMA channel source address value, the DMA shall store the new source address value in the DMA channel SADR.
- If software writes a new DMA channel destination address value, the DMA shall store the new destination address value in the DMA channel DADR.
- **Idle State** (DMA channel CHCSR.TCOUNT!=0) or **Pending State**
  - If software writes a new DMA channel source or destination address value, the DMA shall store the new address value in the DMA channel SHADR.
- **Active State**
  - If software writes a new DMA channel source or destination address value, the DMA shall store the new address value in the ME SHADR.

### End of DMA Transaction

As soon as the DMA completes a DMA transaction, the DMA shall check the value of ME SHADR.

If ME SHADR is not equal to 0000 0000<sub>H</sub>, the ME shall write back data to the DMARAM as follows:

- Shadow Operation Read Only Mode Source Address
  - The DMA shall store the ME SHADR value into DMA channel SADR.
  - The DMA shall store 0000 0000<sub>H</sub> into DMA channel SHADR.
- Shadow Operation Read Only Mode Destination Address
  - The DMA shall store the ME SHADR value into DMA channel DADR.
  - The DMA shall store 0000 0000<sub>H</sub> into DMA channel SHADR.

If software has not executed a shadow address update, ME SHADR is equal to 0000 0000<sub>H</sub>. The ME shall write back data to the DMARAM as follows:

- Shadow Operation Read Only Mode Source Address
  - The DMA shall store 0000 0000<sub>H</sub> into DMA channel SADR (see **Error Conditions**).
- Shadow Operation Read Only Mode Destination Address
  - The DMA shall store 0000 0000<sub>H</sub> into DMA channel DADR (see **Error Conditions**).

### 18.3.4.5.4 Direct Write Mode

If software writes a new DMA channel shadow address value, the DMA shall store the new shadow address value as follows:

- **Idle State** or **Pending State**
  - If software writes a new DMA channel shadow address value, the DMA shall store the new shadow address value in the DMA channel SHADR.
- **Active State**
  - If software writes a new DMA channel shadow address value, the DMA shall store the new shadow address value in the ME SHADR. As soon as the ME shall write back data to the DMA RAM, the DMA shall store the ME SHADR value in the DMA channel SHADR word.

### Start of DMA Transaction

As soon as the DMA services a DMA request, the DMA shall perform a shadow address transfer as follows:

- Shadow Operation Direct Write Mode Source Address
  - If DMA channel SHADR is not 0000 0000<sub>H</sub>, the DMA shall store the DMA channel SHADR value into ME SADR.
- Shadow Operation Direct Write Mode Destination Address

## Direct Memory Access (DMA)

- If DMA channel SHADR is not  $0000\ 0000_H$ , the DMA shall store the DMA channel SHADR value into ME DADR.

### End of Shadow Operation

If software writes a new DMA channel shadow address value of  $0000\ 0000_H$  or applies a **DMA Channel Reset**, the DMA shall not perform a shadow address update at the start of the next DMA transaction.

#### 18.3.4.5.5 Error Conditions

If a DMA channel is configured for **Shadow Operation** and a new DMA request is serviced before software updates the shadow address, the DMA shall perform a DMA move to address  $0000\ 0000_H$  resulting in a bus error.

#### 18.3.4.5.6 Transfer Count Update

The transfer count of a DMA transaction, stored in the DMA channel bit field CHCFGR.TREL, may be programmed if a DMA transaction is running. At the start of a DMA transaction, CHCFGR.TREL is transferred to the ME CHSR.TCOUNT. No reload of address or counter will be done if ME CHSR.TCOUNT is not equal to 0.

The reprogramming of channel specific values (except for the selected shadow address register) must be avoided while a DMA channel is active as the data transfer maybe corrupted.

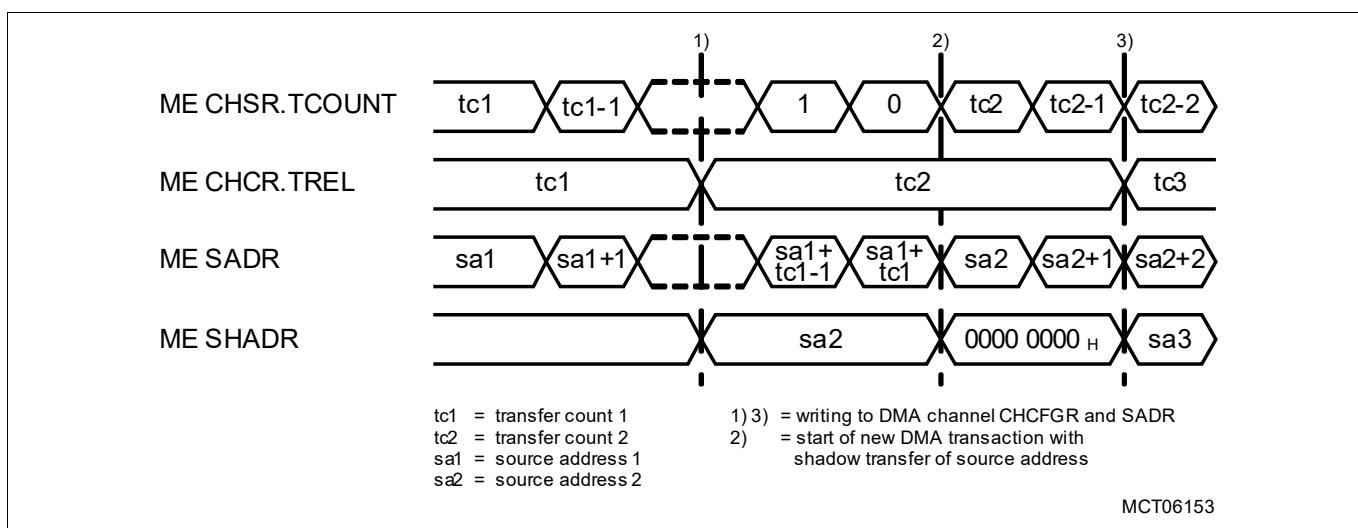
#### Transfer Count and Source Address Update - Example

The contents of the ME transfer count and the ME source address register are updated (at **Figure 204**) during two DMA transactions with a transfer count and a shadowed source address update.

At reference point 2) the DMA transaction 1 is finished and DMA transaction 2 is started. At 1) the DMA channel is reprogrammed with two new parameters for the next DMA transaction: Transfer count tc2 and source address sa2. Source address sa2 is buffered in ME SHADR and transferred to ME SADR when the new DMA transaction is started at 2). At this time, transfer count tc2 is also transferred to ME CHSR.TCOUNT. Note that the shadow address register is only reset by hardware to  $0000\ 0000_H$  as shown in this example, if Read Only Mode is selected.

In the event that DMA channel CHCFGR.TREL is written while DMA channel is active:

- A write to DMA channel CHCFGR.TREL will update the ME CHCR.TREL value.
- On write back DMA channel CHCFGR.TREL is updated with the latest ME CHCR.TREL value.
- ME CHSR.TCOUNT is updated to the new DMA channel CHCFGR.TREL value at the start of the next DMA transaction.



**Figure 204 Shadow Operation Read Only Mode Source Address: Transfer Count and Source Address Update**

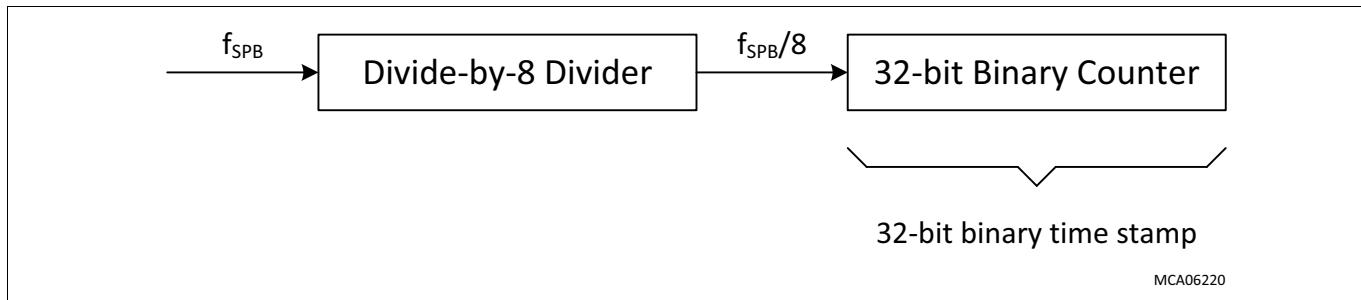
## Direct Memory Access (DMA)

### 18.3.4.6 DMA Timestamp

As soon as the last DMA transfer has completed, a 32-bit DMA timestamp may be appended to record the occurrence of a DMA operation. The DMA timestamp is the time at which the event was recorded by the DMA and not the real time. The DMA transaction completes after the DMA timestamp is written.

#### 18.3.4.6.1 Generation of DMA Timestamp

The DMA timestamp is generated from a 32-bit binary upwards synchronous counter clocked by the SPB clock divided by 8 as shown in [Figure 205](#). The counting starts automatically after the application reset is released. It is not possible to affect the counter during normal DMA operation. The current 32-bit timestamp value may be read through the TIME register.



**Figure 205 Generation of DMA Timestamp**

#### 18.3.4.6.2 Appendage of DMA Timestamp to Non Destination Circular Buffer

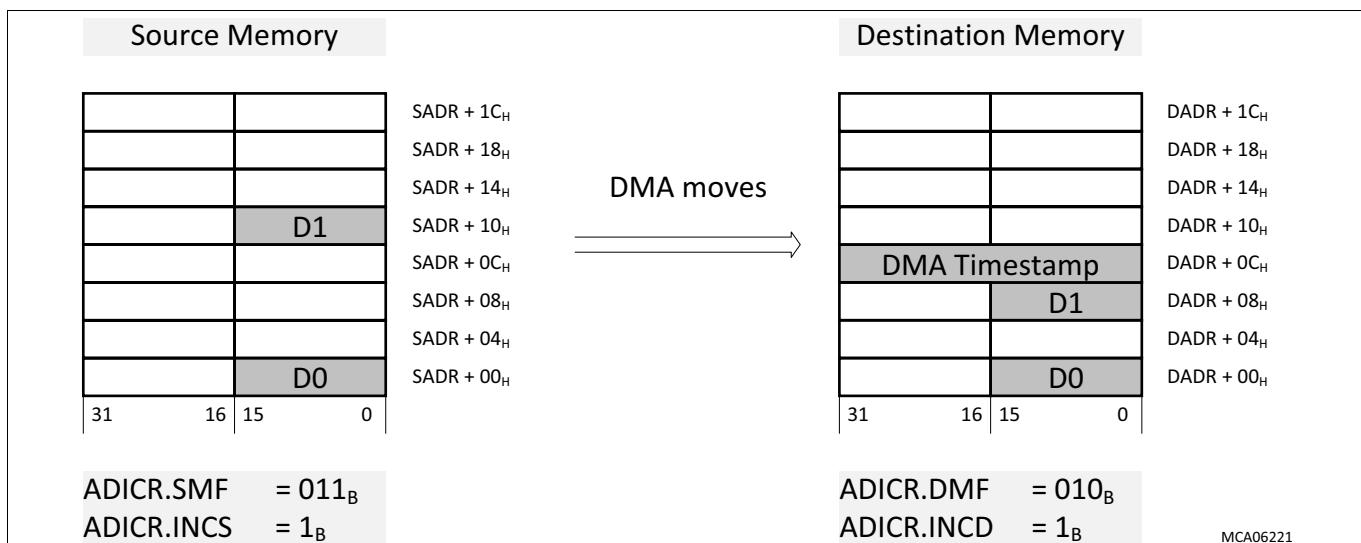
The appendage of a DMA timestamp to the destination data is controlled by the DMA channel ADICR.STAMP bit.

If a DMA channel is configured for the appendage of a DMA timestamp and not destination circular buffer operation (ADICR.DCBE = 0<sub>B</sub>), the ME shall write a DMA timestamp at a destination address defined by:

- If ADICR.INCD = 1<sub>B</sub>, the address of the DMA timestamp is the next higher word aligned destination address.
- If ADICR.INCD = 0<sub>B</sub>, the address of the DMA timestamp is the next lower word aligned destination address.

#### Appendage of DMA Timestamp - Example 1

16-bit half-words are moved from a source memory with an incrementing source address offset of 10<sub>H</sub> to a destination memory with an incrementing destination address offset of 08<sub>H</sub> (at [Figure 206](#)).

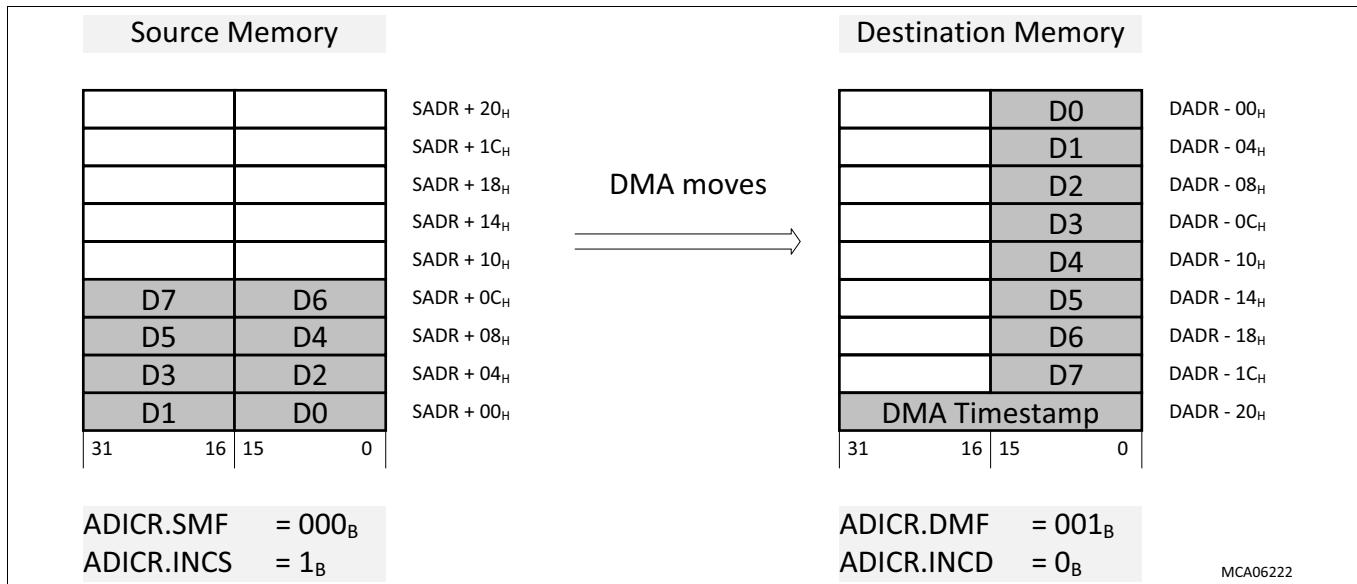


**Figure 206 Incrementing Destination Address: DMA Timestamp appended above DMA Write Move Data**

## Direct Memory Access (DMA)

### Appendage of DMA Timestamp - Example 2

16-bit half-words are moved from a source memory with an incrementing source address offset of  $02_{\text{H}}$  to a destination memory with a decrementing destination address offset of  $04_{\text{H}}$  (at [Figure 207](#)).



**Figure 207 Decrementing Destination Address: DMA Timestamp appended below DMA Write Move Data**

### Destination Address Generation

The destination address of the next DMA write move (of the next DMA transaction) shall be calculated using:

- The destination address of the last DMA write move (of the current DMA transaction).
- The destination address offset defined by ADICR.INCD, ADICR.DMF and CHCFGR.CHDW.

The ME checks the destination address of the next DMA write move.

If the currently calculated destination address of the next DMA write move shall result in DMA write move data overwriting the DMA timestamp of the current DMA transaction, the ME shall calculate a new destination address:

- If DMA channel ADICR.INCD =  $1_{\text{B}}$ , the ME shall select the greater value of the following:
  - Add ( $2_D \times$  destination address offset) to the destination address of last DMA write move.
  - Add  $8_D$  to the destination address of last DMA write move (of the current DMA transaction).
- If DMA channel ADICR.INCD =  $0_{\text{B}}$ , the ME shall select the lesser value of the following:
  - Subtract ( $2_D \times$  destination address offset) from the destination address of last DMA write move.
  - Subtract  $8_D$  from the destination address of last DMA write move (of the current DMA transaction).

The ME shall write back the DMA channel TCS (including updated destination address) to the DMARAM.

### 18.3.4.6.3 Appendage of DMA Timestamp to Destination Circular Buffer

If a DMA channel is configured for the appendage of a DMA timestamp and destination circular buffer operation (ADICR.DCBE =  $1_{\text{B}}$ ), the ME shall write a DMA timestamp at a destination address defined by:

- If ADICR.INCD =  $1_{\text{B}}$  AND ADICR.CBLD = 1 or 0, the address of the DMA timestamp =  $(\text{DADR}[31:2] + 1) \ll 2$
- If ADICR.INCD =  $1_{\text{B}}$  AND ADICR.CBLD > 1, the address of the DMA timestamp =  $(\text{DADR}[31:CBLD] + 1) \ll CBLD$
- If ADICR.INCD =  $0_{\text{B}}$  AND ADICR.CBLD = 1 or 0, the address of the DMA timestamp =  $(\text{DADR}[31:2] \ll 2) - 4$
- If ADICR.INCD =  $0_{\text{B}}$  AND ADICR.CBLD > 1, the address of the DMA timestamp =  $(\text{DADR}[31:CBLD]) \ll CBLD - 4$

The ME writes back to the DMARAM an incremented and wrapped destination address.

## Direct Memory Access (DMA)

### 18.3.4.6.4 Application of DMA Timestamp

The appendage of a DMA timestamp may be used by software to detect DMA failure modes:

- **Lost DMA operation:** DMA request is not serviced (i.e. no data).
- **Unintended DMA operation:** unintentional DMA request results in unexpected DMA transaction (i.e. unexpected data).
- **Delayed DMA transaction:** expected DMA transaction is completed too late.

Software may analyze relative DMA timestamp values to determine the correct sequencing of DMA transactions.

### 18.3.4.7 Pattern Detection

If a DMA channel is configured for **PATDET** then after each DMA read move, the ME compares move data with a value stored in a Pattern Read Register (PRR). The PATDET operation depends on the channel data width (CHCFGR.CHDW). The control field CHCFGR.PATSEL selects different PATDET operations for a specific value of CHDW.

The **Pattern Compare Logic** compares the move data stored in an ME read register to the selected PRR. If the bytes of move data to be compared are spread across different words of move data, the ME may combine the pattern match result of the current **DMA Move** and the pattern match result (stored in ME CHSR.LXO) of the previous **DMA Move**.

#### Configuration

If a DMA channel is configured for **PATDET**:

- The channel data width (CHCFGR.CHDW) shall be configured to 8-bit, 16-bit or 32-bit<sup>1)</sup>.
- The source address shall not be configured to a cached address (segments 8 and 9). A non-cached address (segments A and B) shall be configured to prevent the **ME Read Buffer** re-aligning the move data.

#### Interrupt Service Request

If a DMA channel is configured for PATDET and a pattern match is detected:

- The DMA transaction ends with the current DMA move.
- DMA channel TSR.HTRE is cleared to stop DMA transfers in the current active DMA channel.
- DMA channel TSR.CH will clear when the DMA write move has completed.
- The ME shall write back the next DMA move values of SADR and DADR.
  - If the DMA channel is configured for **Shadow Operation Read Only Mode**, the ME shall not perform a shadow address update. The ME write backs the current value of SHADR to the DMARAM.
- A PATDET interrupt service request shall be triggered.

Software may read the DMA channel status to identify the location of the pattern match in the data stream.

#### Errors

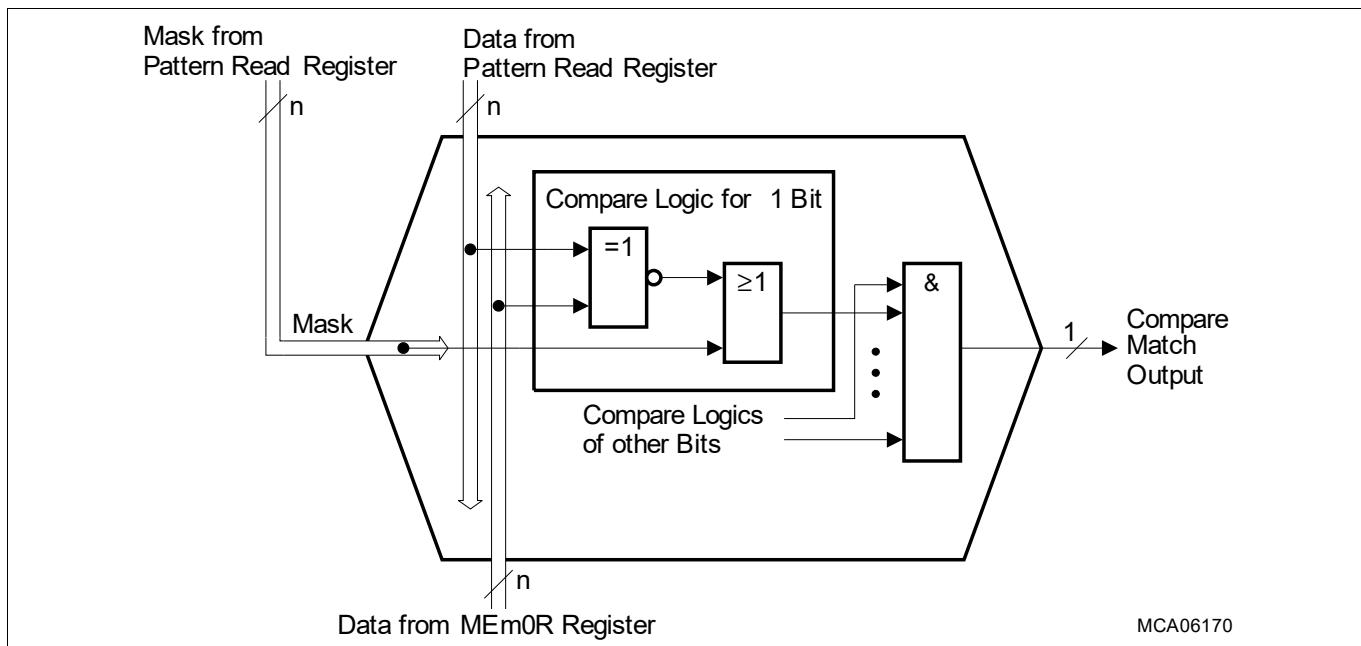
If a **SER** is reported during the current DMA read move, the ME shall not perform a pattern match.

### 18.3.4.7.1 Pattern Compare Logic

The ME COMPare (COMP) logic (at **Figure 208**) compares move data against a reference pattern at a bit-wise level. One COMP block controls either 8 bits or 16 bits of data. The COMP block may be configured to mask each data bit for the pattern compare.

<sup>1)</sup> If a DMA channel is configured for PATDET, the ME shall only store move data in MEm1R and/or MEm0R.

## Direct Memory Access (DMA)



**Figure 208 Pattern Compare Logic (COMP Block)**

In the COMP logic, one data bit from a ME read register is compared to the corresponding pattern bit stored in the selected PRR. If both bits are equal and a pattern mask bit stored in another PRR bit field is  $0_B$  then the compare matched condition becomes active. When the pattern mask bit is set to  $1_B$ , the compare matched condition is always active (set) for the related bit. When the compare matched conditions for each bit within the COMP logic are true, the compare match output line of the COMP block becomes active.

### 18.3.4.7.2 Pattern Detection for 8-bit Channel Data Width

If a DMA channel is configured for **Pattern Detection for 8-bit Channel Data Width**, the **ME** shall use the source address to select byte(s) of move data to be compared with patterns stored in PRR0 or PRR1. Three pattern compare configurations are possible defined by the DMA channel pattern select (CHCFG.R.PATSEL[1:0]).

The pattern compare logic allows a byte of DMA move data to be compared with two different patterns.

A mask operation of each compared bit is possible.

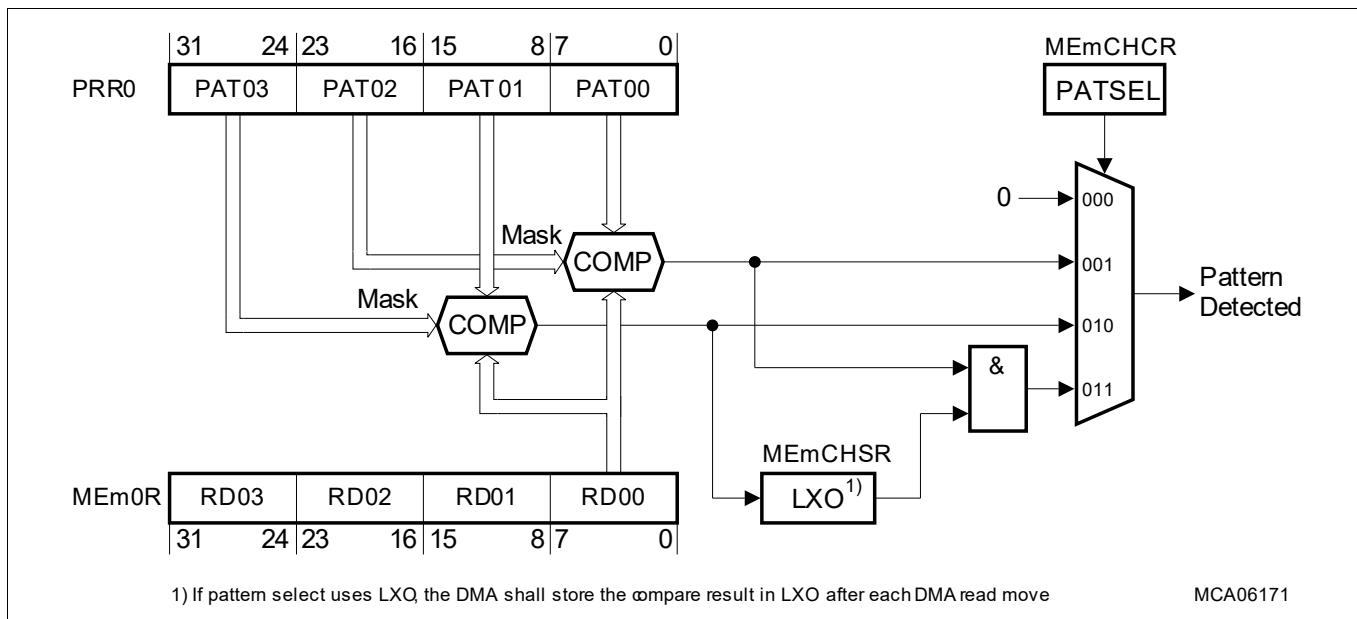
#### Example of PATDET for 8-bit Channel Data Width

The DMA channel (at **Figure 209**) is configured for PATDET for 8-bit channel data width (CHCFG.R.CHDW =  $000_B$ ).

If the current DMA read move stores the move data byte in MEm0R.RD00 then

- The **ME** uses the source address to select the move data byte stored in MEm0R.RD00.
- The **ME** compares move data byte stored in MEm0R.RD00 to PRR0.

## Direct Memory Access (DMA)



**Figure 209** PATDET for 8-bit Channel Data Width and Move Data Byte stored in MEm0R.RD00

The DMA channel may be configured for one of three pattern compare operations (at [Table 577](#)).

**Table 577** PATDET for 8-bit Channel Data Width and Move Data Byte stored in MEm0R.RD00

PATSE L	Pattern Detection Operating Modes
000 <sub>B</sub>	Pattern detection disabled
001 <sub>B</sub>	Pattern compare of MEm0R.RD00 to PRR0.PAT00, masked by PRR0.PAT02
010 <sub>B</sub>	Pattern compare of MEm0R.RD00 to PRR0.PAT01, masked by PRR0.PAT03
011 <sub>B</sub>	Pattern compare of MEm0R.RD00 to PRR0.PAT00, masked by PRR0.PAT02 of the <u>current</u> DMA read move <b>and</b> Pattern compare of MEm0R.RD00 to PRR0.PAT01, masked by PRR0.PAT03 of the <u>previous</u> DMA read move
100 <sub>B</sub>	Pattern detection disabled
101 <sub>B</sub>	Pattern compare of MEm0R.RD00 to PRR1.PAT10, masked by PRR1.PAT12
110 <sub>B</sub>	Pattern compare of MEm0R.RD00 to PRR1.PAT11, masked by PRR1.PAT13
111 <sub>B</sub>	Pattern compare of MEm0R.RD00 to PRR1.PAT10, masked by PRR1.PAT12 of the <u>current</u> DMA read move <b>and</b> Pattern compare of MEm0R.RD00 to PRR1.PAT11, masked by PRR1.PAT13 of the <u>previous</u> DMA read move

### Example of Two Byte PATDET Sequence

If a DMA channel is configured for a two byte PATDET sequence compared to PRR0 (CHCFGR.PATSEL[2:0] = 011<sub>B</sub>) then after each DMA move the pattern match result of move data compared with PRR0.PAT01, masked by PRR0.PAT03 is stored in bit MEmCHSR.LXO.

This DMA channel configuration allows, for example, two-byte sequences to be detected in an 8-bit data stream coming from a serial peripheral unit with 8-bit data width (e.g. recognition of carriage-return, line-feed characters, etc.).

## Direct Memory Access (DMA)

### 18.3.4.7.3 Pattern Detection for 16-bit Channel Data Width

If a DMA channel is configured for **Pattern Detection for 16-bit Channel Data Width**, the **ME** shall use the source address to select bytes of move data to be compared with patterns stored in PRR0 or PRR1. Three pattern match configurations are possible defined by the DMA channel pattern select (CHCFGFR.PATSEL[1:0]):

- Aligned Mode: compare complete half-word of one DMA move only.
- Unaligned Mode: compare upper and lower byte of two consecutive DMA moves.
- Combined Mode: combine Aligned Mode and Unaligned Mode.

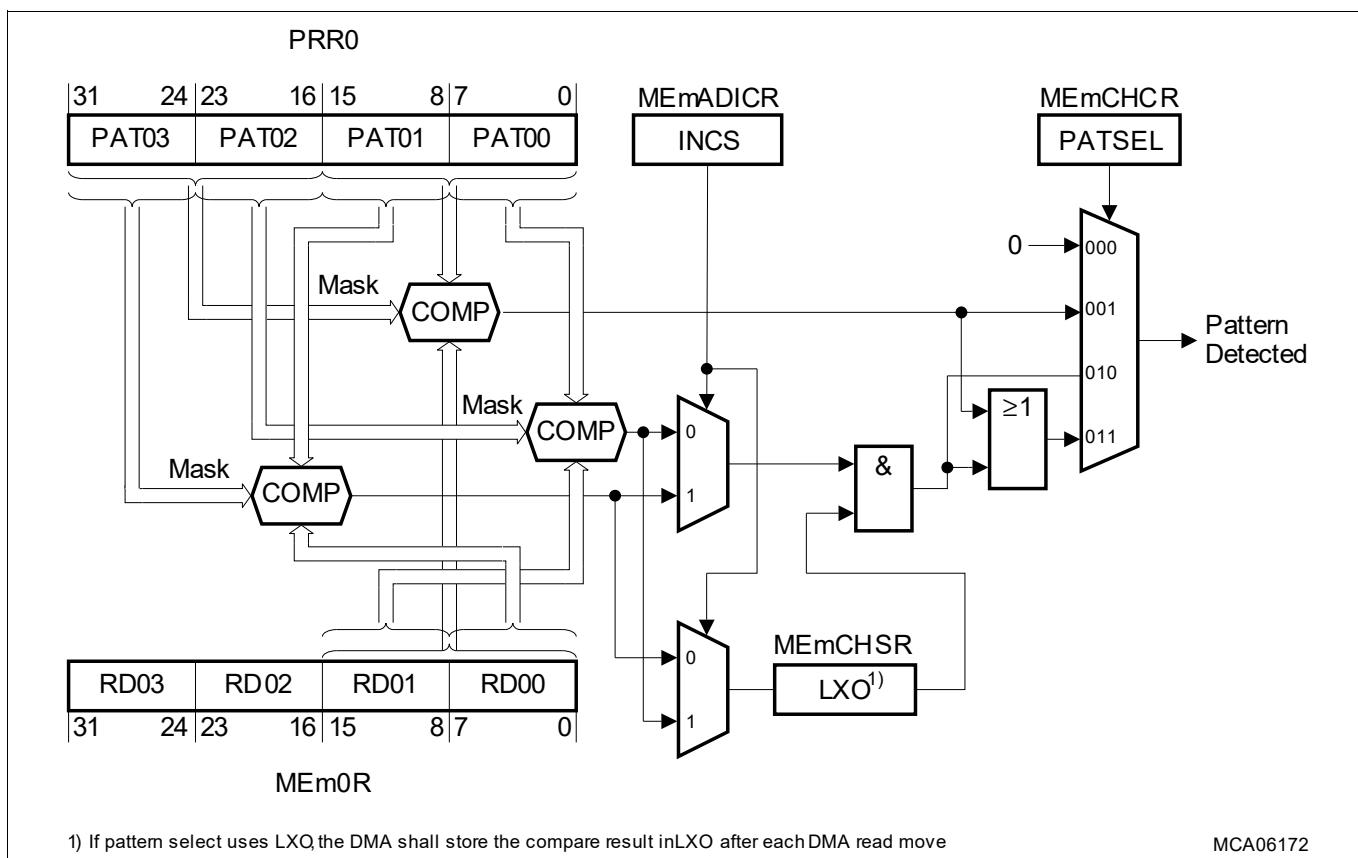
A mask operation of each compared bit is possible.

#### Example of PATDET for 16-bit Channel Data Width

The DMA channel (at [Figure 210](#)) is configured for PATDET for 16-bit channel data width (CHCFG.RCHD = 001<sub>B</sub>).

If the current DMA read move stores the move data half word in MEm0R.RD01||RD00 then

- The **ME** uses the source address to select the move data half word stored in MEm0R.RD01||RD00.
- The **ME** compares move data half word stored in MEm0R.RD01||RD00 to PRR0.



**Figure 210** PATDET for 16-bit Channel Data Width and Move Data stored in MEm0R.RD01||RD00

The DMA channel may be configured for one of three pattern compare operations (at [Table 578](#)).

- Aligned Mode
- Unaligned Mode 1 (Source Address Decrement)
  - The high byte of the current 16-bit DMA read move and the low byte of the previous 16-bit DMA read move are compared.
- Unaligned Mode 2 (Source Address Increment)

## Direct Memory Access (DMA)

- The low byte of the current 16-bit DMA read move and the high byte of the previous 16-bit DMA read move are compared.
- Combined Mode
  - If it is not known on which byte boundary (even or odd address) the 16-bit pattern to be detected is located, the DMA channel should be configured for combined mode.
  - The combined mode is the most flexible mode that combines the pattern search capability for aligned and unaligned 16-bit data searches.

**Table 578 PATDET for 16-bit Channel Data Width and Move Data stored in MEm0R.RD01||RD00**

PATSE L	INCS	Pattern Detection Operating Modes
000 <sub>B</sub>	-	Pattern detection disabled
001 <sub>B</sub>	-	<b>Aligned Mode</b> Pattern compare of MEm0R.RD01  RD00 to PRR0.PAT01  PAT00, masked by PRR0.PAT03  PAT02
010 <sub>B</sub>	0	<b>Unaligned Mode 1 (Source Address Decrement)</b> Pattern compare of MEm0R.RD01 to PRR0.PAT00, masked by PRR0.PAT02 of the <u>current</u> DMA read move <b>and</b> Pattern compare of MEm0R.RD00 to PRR0.PAT01, masked by PRR0.PAT03 (LXO) of the <u>previous</u> DMA read move
	1	<b>Unaligned Mode 2 (Source Address Increment)</b> Pattern compare of MEm0R.RD00 to PRR0.PAT01, masked by PRR0.PAT03 of the <u>current</u> DMA read move <b>and</b> Pattern compare of MEm0R.RD01 to PRR0.PAT00, masked by PRR0.PAT02 (LXO) of the <u>previous</u> DMA read move
011 <sub>B</sub>	0 or 1	<b>Combined Mode</b> Pattern compare for aligned mode (PATSEL = 001 <sub>B</sub> ) <b>or</b> unaligned modes (PATSEL = 010 <sub>B</sub> )
100 <sub>B</sub>	-	Pattern detection disabled
101 <sub>B</sub>	-	<b>Aligned Mode</b> Pattern compare of MEm0R.RD01  RD00 to PRR1.PAT11  PAT10, masked by PRR1.PAT13  PAT12
110 <sub>B</sub>	0	<b>Unaligned Mode 1 (Source Address Decrement)</b> Pattern compare of MEm0R.RD01 to PRR1.PAT10, masked by PRR1.PAT12 of the <u>current</u> DMA read move <b>and</b> Pattern compare of MEm0R.RD00 to PRR1.PAT11, masked by PRR1.PAT13 (LXO) of the <u>previous</u> DMA read move
	1	<b>Unaligned Mode 2 (Source Address Increment)</b> Pattern compare of MEm0R.RD00 to PRR1.PAT11, masked by PRR1.PAT13 of the <u>current</u> DMA read move <b>and</b> Pattern compare of MEm0R.RD01 to PRR1.PAT10, masked by PRR1.PAT12 (LXO) of the <u>previous</u> DMA read move
111 <sub>B</sub>	0 or 1	<b>Combined Mode</b> Pattern compare for aligned mode (PATSEL = 101 <sub>B</sub> ) <b>or</b> unaligned modes (PATSEL = 110 <sub>B</sub> )

## Direct Memory Access (DMA)

### 18.3.4.7.4 Pattern Detection for 32-bit Channel Data Width

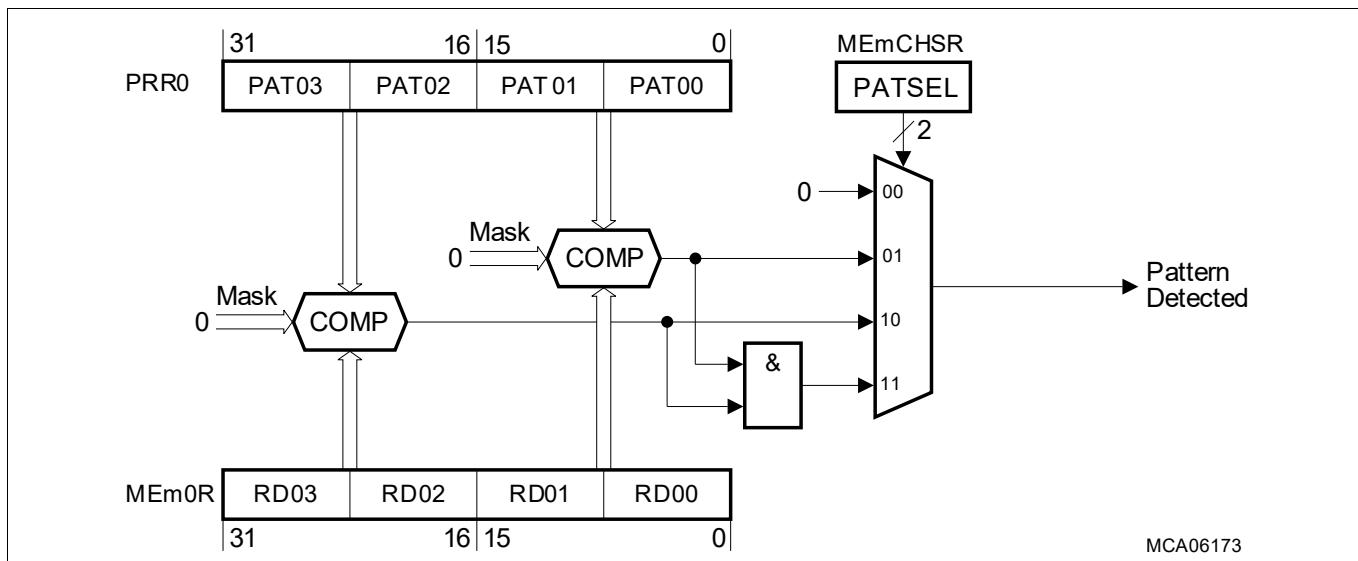
If a DMA channel is configured for **Pattern Detection for 32-bit Channel Data Width**, the **ME** shall compare a move data word with patterns stored in PRR0 or PRR1. Three pattern compare configurations are possible defined by the DMA channel pattern select (CHCFGFR.PATSEL[1:0]). A mask operation is not possible.

#### Example of PATDET for 32-bit Channel Data Width

The DMA channel (at [Figure 211](#)) is configured for PATDET and 32-bit channel data width (CHCFG.R.CHDW = 010<sub>B</sub>).

If the current DMA read move stores the move data word in MEm0R then

- The **ME** uses the source address to select the move data word stored in MEm0R.
- The **ME** compares move data word stored in MEm0R to PRR0.



**Figure 211** PATDET for 32-bit Channel Data Width and Move Data stored in MEm0R

The DMA channel may be configured for one of three pattern compare operations (at [Table 579](#)).

- Lower half-word only compared to pattern stored in PRR0 or PRR1.
- Upper half-word only compared to pattern stored in PRR0 or PRR1.
- Complete 32-bit word compared to pattern stored in PRR0 or PRR1.

**Table 579** PATDET for 32-bit Channel Data Width and Move Data stored in MEm0R

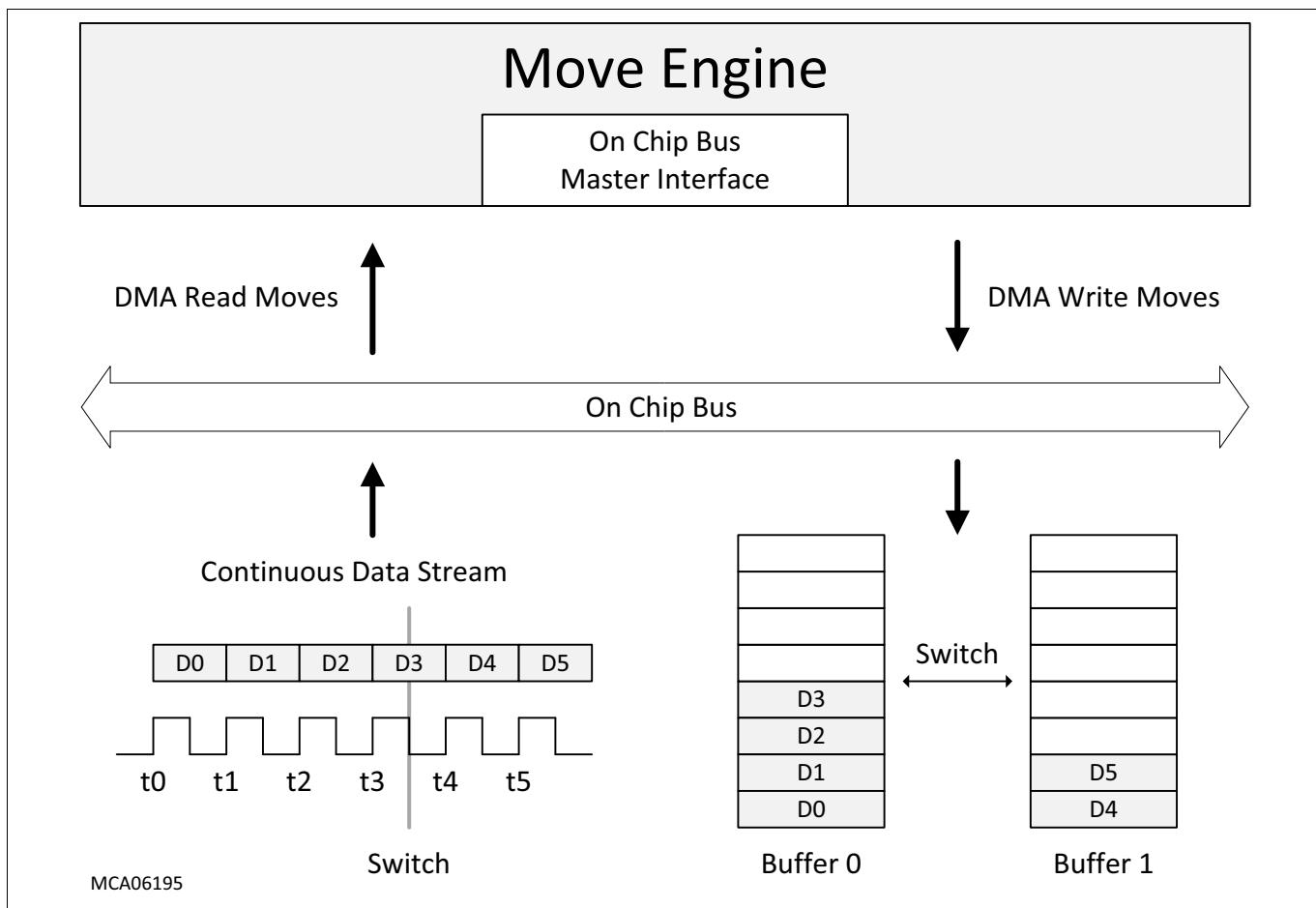
PATSEL	Pattern Detection Operating Modes
000 <sub>B</sub>	Pattern detection disabled
001 <sub>B</sub>	Unmasked pattern compare of MEm0R.RD01  RD00 to PRR0.PAT01  PAT00
010 <sub>B</sub>	Unmasked pattern compare of MEm0R.RD03  RD02 to PRR0.PAT03  PAT02
011 <sub>B</sub>	Unmasked pattern compare of MEm0R.RD03  RD02  RD01  RD00 to PRR0.PAT03  PAT02  PAT01  PAT00
100 <sub>B</sub>	Pattern detection disabled
101 <sub>B</sub>	Unmasked pattern compare of MEm0R.RD01  RD00 to PRR1.PAT11  PAT10
110 <sub>B</sub>	Unmasked pattern compare of MEm0R.RD03  RD02 to PRR1.PAT13  PAT12
111 <sub>B</sub>	Unmasked pattern compare of MEm0R.RD03  RD02  RD01  RD00 to PRR1.PAT13  PAT12  PAT11  PAT10

## Direct Memory Access (DMA)

### 18.3.4.8 Double Buffering Operations

The DMA supports double buffering. For example, Double Destination Buffering (at [Figure 212](#)) is as follows:

- DMA read moves transfer a continuous data stream from a peripheral to the DMA.
  - DMA write moves transfer the read data from the DMA to one of two destination buffers stored in memory.
- The dormant buffer is frozen and available for cyclic software tasks while the other buffer continues to be filled.



**Figure 212 DMA Double Buffering**

If a DMA channel is configured for one of the [Double Buffering Operations](#) AND is triggered by [DMA Hardware Request](#), software should configure the DMA channel for [Continuous Mode](#) (DMA channel CHCFGR.CHMODE = 1<sub>B</sub>).

#### 18.3.4.8.1 DMA Double Source Buffering

Data is streamed by the DMA from one of two data buffers (buffer 0 and buffer 1) stored in memory to a continuous output data stream. The Double Source Buffering addresses are:

- The source address SADR shall address source buffer 0.
- The shadow address SHADR shall address source buffer 1.

#### 18.3.4.8.2 DMA Double Destination Buffering

A continuous input data stream is directed by the DMA to one of two data buffers (buffer 0 and buffer 1) stored in memory. The Double Destination Buffering addresses are:

- The destination address DADR shall address destination buffer 0.

## Direct Memory Access (DMA)

- The shadow address SHADR shall address destination buffer 1.

### 18.3.4.8.3 Size of Buffer

The data size of the two source/destination buffers is equal and determined by the following TCS parameters:

- The channel data width (CHCFG.R.CHDW).
- The block mode value (CHCFG.R.BLKM).
- The transfer reload value (CHCFG.R.TREL).

The memory size of the two source/destination buffers is determined by the additional TCS parameters:

- The source/destination circular buffer enable/disable control (ADICR.SCBE/DCBE).
- The source/destination address modification factor (ADICR.DMF/SMF).
- The increment of the source/destination address (ADICR.INCD/INCS).

The size of one buffer is equal to the size of one DMA transaction.

### 18.3.4.8.4 Buffer Switch

The re-direction of the data stream from one buffer to the other buffer shall be controlled by a **Software Switch** and additionally **Automatic Hardware Switch**. During a buffer switch:

- No DMA requests shall be lost by the DMA.
- There shall be no loss, duplication or split of data across the two buffers.

#### Active Buffer Status

During DMA double buffering the DMA channel CHCSR.BUFFER bit is used to indicate which buffer is active.

- DMA channel CHCSR.BUFFER = 0<sub>B</sub>; buffer 0 is read or filled by the DMA.
- DMA channel CHCSR.BUFFER = 1<sub>B</sub>; buffer 1 is read or filled by the DMA.

#### Frozen Buffer Status

When the DMA switches buffers the DMA channel frozen bit is set (CHSR.FROZEN= 1<sub>B</sub>). The frozen buffer is then available for a cyclic software task. On completion of the software task the status bit CHSR.FROZEN is cleared by software and the buffer address pointer is re-programmed to the start address ready for the next DMA transaction that reads/fills the buffer.

#### Active Buffer Empty or Full

If the active buffer is emptied (in the case of DMA double source buffering) or filled (in the case of DMA double destination buffering) and no automatic hardware switch occurs before a software buffer switch is received then the DMA channel will stop and no more DMA transfers are made. When the DMA channel stops, the transfer count (CHCSR.TCOUNT) is equal to 0000<sub>H</sub>.

A buffer empty or full event may be signalled by a **DMA Channel Interrupt Service Request**. The DMA channel interrupt control ADICR.INTCT should be configured to 10<sub>B</sub> and the interrupt raise detect value ADICR.IRDV to 0000<sub>B</sub>. When the transfer count equals zero then the DMA channel will raise a DMA channel interrupt service request.

The DMA channel interrupt handler can interrogate the DMA to identify which DMA channel generated the interrupt service request. Software may initialize the DMA channel ready to restart double buffering operations.

#### Active Buffer Overflow

If the data rate is faster than the DMA is able to transfer the data to one of the buffers then a **TRL** event shall occur.

## Direct Memory Access (DMA)

### 18.3.4.8.5 Software Switch

Before a DMA transaction completes and the active buffer is full software shall re-direct the data stream from the active buffer to the other buffer by setting the DMA channel CHCSR.SWB. If a ME is actively servicing a DMA channel configured for double buffering then the current DMA transfer shall complete before the buffer switch is made. On completion of the DMA transfer the DMA shall:

- Automatically re-load DMA channel CHCSR.TCOUNT with CHCFGR.TREL
- Switch the address pointer:
  - Buffer 0 address pointer (source address or destination address).
  - Buffer 1 address pointer (shadow address).

The TCS address control factors remain the same. The next DMA transaction controlling the filling or reading of the new buffer will start when a DMA request is received.

If software re-directs the data stream by setting DMA channel CHCSR.SWB = 1<sub>B</sub>, software shall poll DMA channel CHCSR.FROZEN AND DMA channel CHCSR.BUFFER. If software reads DMA channel CHCSR.FROZEN = 1<sub>B</sub> and an active buffer transition, software may start to analyze the frozen buffer.

If a DMA channel is configured for Software Switch Only AND the DMA transaction is completed before the DMA receives a Software Switch, the DMA shall report a **TRL** event (DMA channel TSR.TRL = 1<sub>B</sub>). If the DMA channel enable TRL bit is set (TSR.ETRL = 1<sub>B</sub>), the DMA shall trigger a **DMA RP Error Interrupt Service Request**.

### 18.3.4.8.6 Automatic Hardware Switch

The DMA will automatically switch from the active buffer to the other buffer when the DMA transaction reading or filling the active buffer completes. On switching buffers the DMA channel CHCSR.FROZEN bit is set to indicate that one buffer is frozen and available for cyclic software tasks. The completion of a DMA transaction may be signalled by a **DMA Channel Interrupt Service Request**. On completion of the DMA transaction when ME CHSR.TCOUNT equals 0000<sub>H</sub> the DMA shall:

- Automatically re-load DMA channel CHCSR.TCOUNT with CHCFGR.TREL
- Switch the address pointer to select the other buffer:
  - Buffer 0 address pointer (source address or destination address).
  - Buffer 1 address pointer (shadow address).

The DMA channel TCS address control factors remain the same. The next DMA transaction controlling the filling or reading of the new buffer will start when a DMA request is received.

If DMA channel CHCSR.FROZEN is equal to 1<sub>B</sub>, the automatic hardware switch will not occur and the DMA shall report a **TRL** event (DMA channel TSR.TRL = 1<sub>B</sub>). If the DMA channel enable TRL bit is set (TSR.ETRL = 1<sub>B</sub>), the DMA shall trigger a **DMA RP Error Interrupt Service Request**.

### 18.3.4.8.7 Application of Double Buffering

A typical DMA double buffer application is shown in [Figure 213](#):

- DMA channel m is configured for **DMA Double Destination Buffering**. The DMA channel moves the input data generated by a sensor into the active destination buffer.
- A cyclic software task processes the input data stored in the frozen destination buffer and writes the output into the frozen source buffer. The CPU controls both the source and destination buffer switching.
- DMA channel n is configured for **DMA Double Source Buffering**. The DMA channel moves the processed data from the frozen source buffer to the actuator.

The system configuration shall balance the input data rate, the duration of cyclic software tasks and the buffer size in order to prevent **TRL** events.

### Direct Memory Access (DMA)

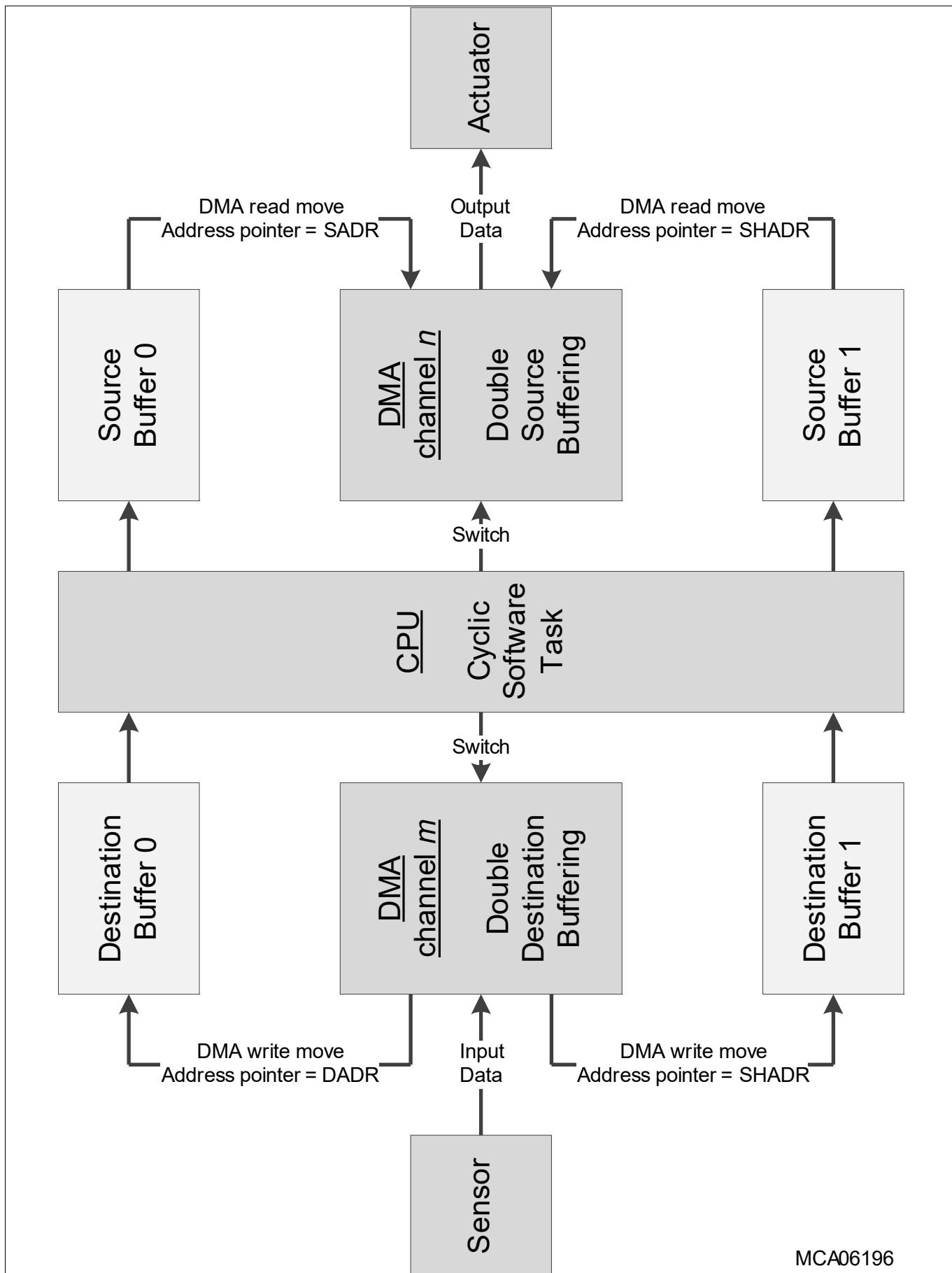


Figure 213 Application of Double Buffering

## Direct Memory Access (DMA)

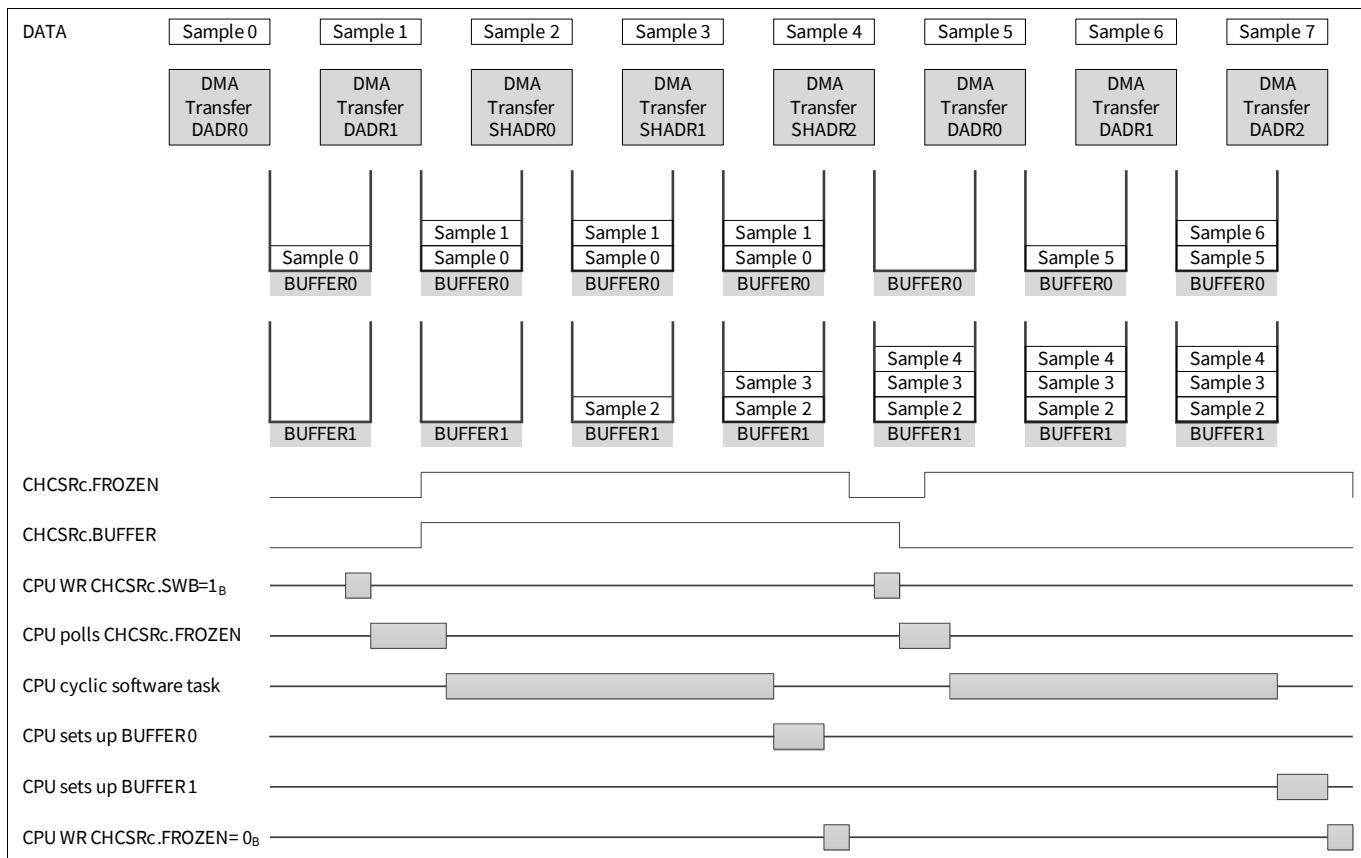


Figure 214 DMA Double Destination Buffering Software Switch Only

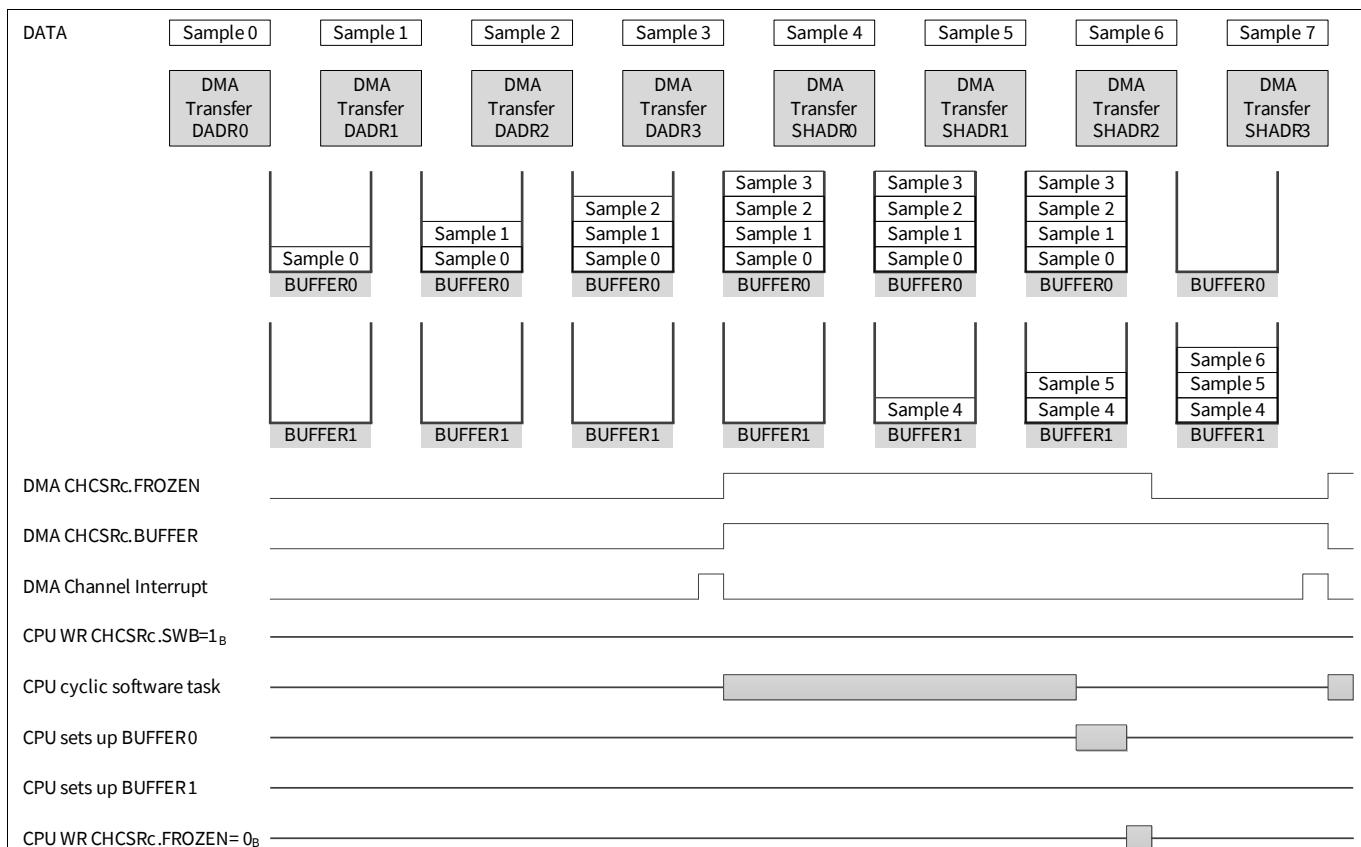


Figure 215 DMA Double Destination Buffering Software Switch and Automatic Hardware Switch

## Direct Memory Access (DMA)

### 18.3.4.9 Linked List Operations

Linked list operations are an extension of DMA channel functionality to support the more flexible use of the DMA. A linked list operation shall consist of a series of DMA transactions executed by the same DMA channel. Each DMA transaction shall have a unique TCS. The source and destination areas shall not have to exist in contiguous areas of memory.

If a DMA channel is configured for linked list operation then as soon as the ME completes the current DMA transaction, the ME shall read the next TCS from memory and overwrite the current TCS at the DMA channel location in DMARAM. The current DMA transaction uses a 32-byte aligned address pointer to point to the next TCS stored in either internal or external memory. The DMA shall start reading the next TCS from word 0 upwards. There is no limit to the number of DMA transactions in a linked list operation. The first DMA transaction in a linked list operation shall be initiated by a DMA hardware request or a DMA software request. Subsequent DMA transactions may additionally be initiated by a **DMA Auto Start Request**.

Each DMA channel supports the following types of linked list operation:

- **DMA Linked List (DMALL)**
- **Accumulated Linked List (ACLLL)**
- **Safe Linked List (SAFLL)**
- **Conditional Linked List (CONLL)**

#### 18.3.4.9.1 DMA Auto Start Request

If a **DMA Auto Start Request** is selected in the next TCS (CHCSR.SCH = 1<sub>B</sub>) then the next DMA transaction shall be initiated by the ME. The access pending bit TSR.CH shall be set. The DMA shall perform an **Arbitration Sequence** to determine the highest priority DMA request. A **DMA Auto Start Request** shall bypass the Interrupt Router so reducing the cumulative latency over a number of DMA transactions.

#### 18.3.4.9.2 Non Linked List Operation

If the next TCS is not configured for linked list operation, the next TCS should be configured for a **Move Operation**. The DMA channel exits linked list operation. A **DMA Auto Start Request** shall not initiate the next DMA transaction.

#### 18.3.4.9.3 Last DMA Transaction

A **DMA Channel Interrupt Service Request** may be used to signal the completion of the last DMA transaction in a sequence of linked list operations. The last DMA transaction will load the next TCS. **DMA Auto Start Request** shall not be selected.

#### 18.3.4.9.4 Circular Linked List Operations

A sequence of linked list operations may be configured for circular operation. The same series of DMA transactions shall be repeated.

## Direct Memory Access (DMA)

### 18.3.4.9.5 DMA Linked List (DMALL)

The DMA channel (at [Figure 216](#)) is configured for DMALL operation. The DMA channel shadow address register (SHADR) stores the 32-bit address pointer to the next TCS. As soon as the current DMA transaction completes, the ME reads the next TCS and stores it in DMARAM.

#### DMA Address Checksum & DMA Data Checksum

Software shall configure the initial value of **DMA Address Checksum** and **DMA Data Checksum** used during the first DMA transaction. When the ME reads the next TCS, the **DMA Address Checksum** and **DMA Data Checksum** shall be initialized to  $00000000_H$ .

### 18.3.4.9.6 Accumulated Linked List (ACCLL)

ACCLL is a variant of DMALL and has an identical footprint in memory (size and structure). The SDCRC and RDCRC checksums are accumulated across DMA transactions.

#### DMA Address Checksum & DMA Data Checksum

The **DMA Address Checksum** and **DMA Data Checksum** are not overwritten when the next TCS is loaded into the DMA channel. As long as the DMA channel is configured for ACCLL, the **DMA Address Checksum** and **DMA Data Checksum** are calculated across all DMA transactions.

### 18.3.4.9.7 Safe Linked List (SAFLL)

The DMA channel (at [Figure 217](#)) is configured for SAFLL operation. SAFLL provides protection against software errors. If the address pointer in a sequence of linked list operations is incorrectly written then the address pointer may point to any random area of memory and load the next TCS from the wrong location and execute it. As long as a DMA channel is configured for SAFLL, the ME checks that the calculated **DMA Address Checksum** matches an expected **DMA Address Checksum** stored in the next TCS.

#### DMA Address Checksum

The user is required to load the SDCRC word with an expected **DMA Address Checksum** value. As soon as the ME has read the next TCS from memory, the ME compares the **DMA Address Checksum** calculated by the current DMA transaction against the expected **DMA Address Checksum** stored in the next TCS. If the checksums match then the DMA proceeds with the execution of the next TCS. If the checksums do not match then the ME records a **SAFLL DMA Address Checksum Error** and triggers a **DMA RP Error Interrupt Service Request**. The DMA stops the execution of the linked list operation. Assuming all the **DMA Address Checksum** values match during the sequence of linked list operations then the **DMA Address Checksum** is calculated across all DMA transactions.

#### DMA Data Checksum

The **DMA Data Checksum** is not overwritten when the new TCS is loaded into the DMA channel. The **DMA Data Checksum** is calculated across all DMA transactions.

## Direct Memory Access (DMA)

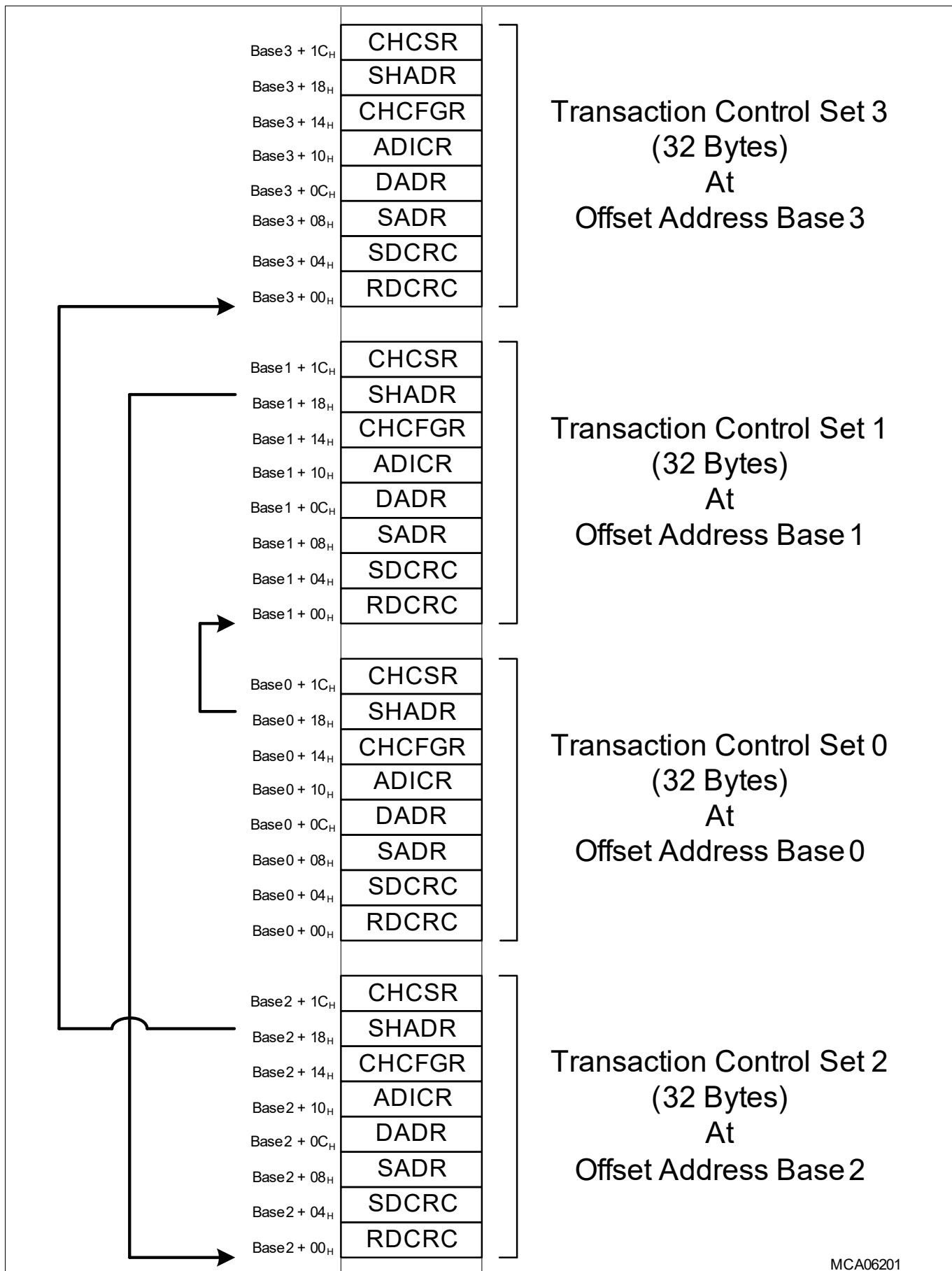


Figure 216 DMA Linked List

## Direct Memory Access (DMA)

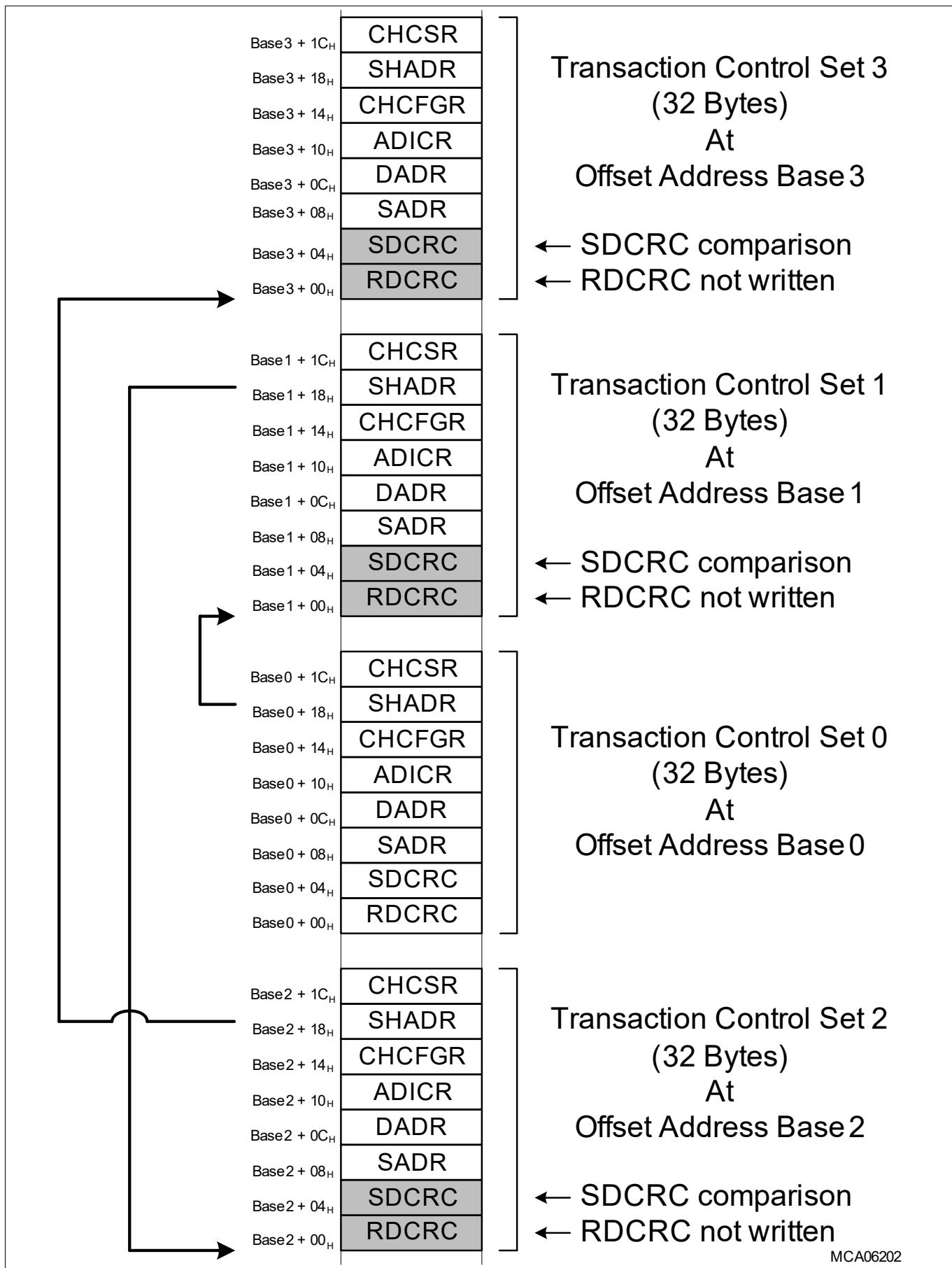


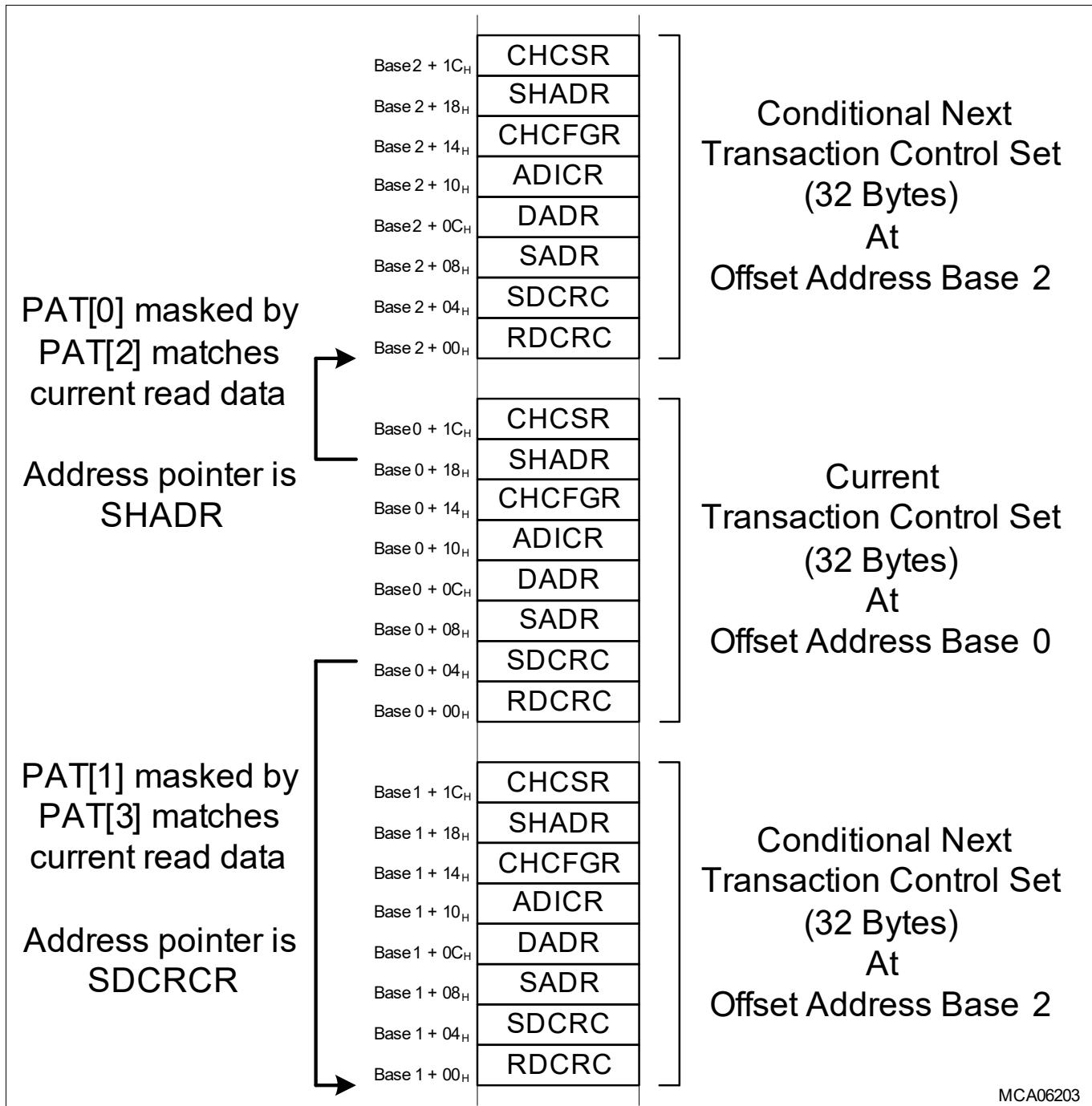
Figure 217 Safe Linked List

## Direct Memory Access (DMA)

### 18.3.4.9.8 Conditional Linked List (CONLL)

A special use of a linked list is CONLL (at [Figure 218](#)). Selection of the address pointer to the next TCS is determined by a conditional state. CONLL uses the shadow address (SHADR) and the source and destination address CRC registers (SDCRCR) as address pointers.

A CONLL operation is limited to 8-bit DMA moves ( $\text{CHCFGR.CHDW} = 000_B$ ). The source address shall not be configured to a cached address (segments 8 and 9). A non-cached address (segments A and B) shall be configured to prevent the [ME Read Buffer](#) re-aligning the move data.

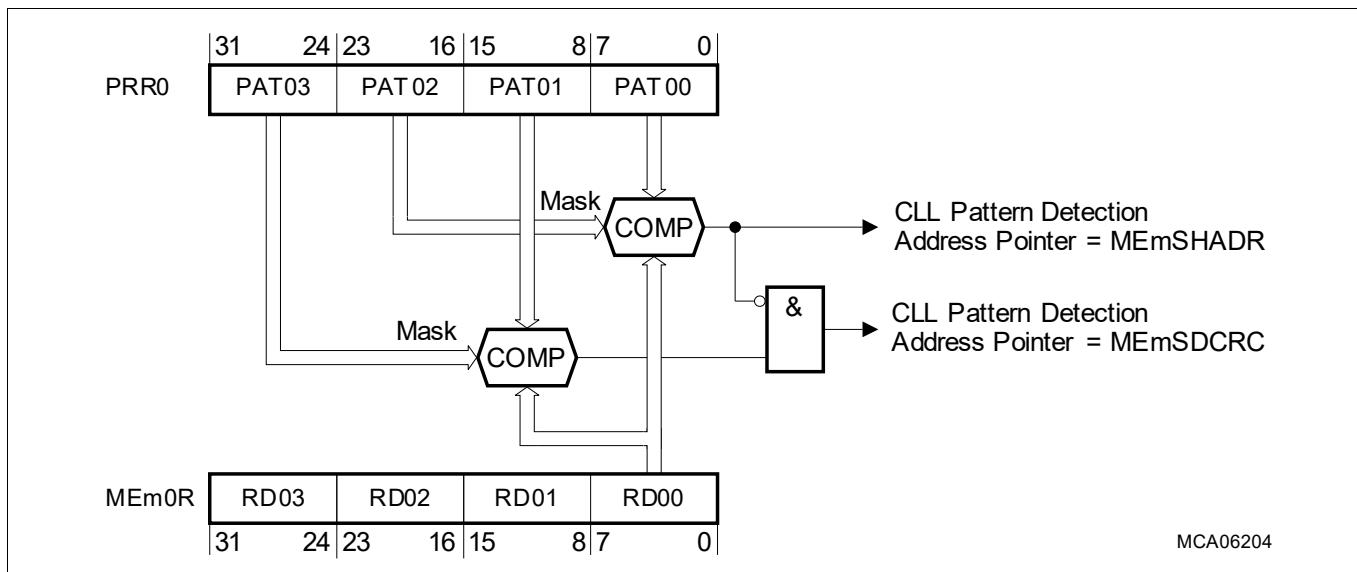


**Figure 218** Conditional Linked List

#### CONLL Pattern Detection

The DMA channel (at [Figure 219](#)) is configured for CONLL. The TCS selects PRR0 for a pattern compare.

## Direct Memory Access (DMA)



**Figure 219 Conditional Linked List and Pattern Compare Logic**

During each DMA move the pattern detection logic determines the selection of the address pointer:

- If PAT00 masked by PAT02 matches the DMA move data stored in MEm0R RD00 then as soon as the DMA move completes the ME shall load the DMA channel with the next TCS read from an address stored in SHADR.
- If PAT01 masked by PAT03 matches the DMA move data stored in MEm0R RD00 then as soon as the DMA move completes the ME shall load the DMA channel with the next TCS read from an address stored in SDCRCR.
- If both PAT00 masked by PAT02 and PAT01 masked by PAT03 match the DMA move data stored in MEm0R RD00 then as soon as the ME shall load the DMA channel with the next TCS read from an address stored in SHADR.
- If there is no pattern match then the ME continues with the DMA transaction.

If a pattern match is detected then:

- The DMA transaction ends with the current DMA move.
- The DMA channel TSR.CH will be cleared when the DMA write move has completed.

### Completion of Current DMA Transaction

If the current DMA transaction completes when ME CHSR.TCOUNT = 0<sub>D</sub> and there is no pattern match then the CONLL will load the DMA channel with the next TCS read from an address stored in SHADR.

### Multiple Pattern Detection Conditions

The user may intentionally program PAT00 masked by PAT02 shall not match with the DMA move data stored in MEm0R RD00 and transition across a series of DMA transactions in the CONLL. For each DMA transaction the user may programme a series of different PAT01 masked by PAT03 values and test if each value matches with the current DMA move data stored in MEm0R RD00. If PAT01 masked by PAT03 matches the current DMA move data stored in MEm0R RD00, then as soon as the DMA move is completed, the ME shall load the DMA channel with the next TCS read from an address stored in SDCRCR.

## Direct Memory Access (DMA)

### 18.3.4.10 DMA Data Checksum

To support enhanced data integrity checking the DMA calculates a 32-bit Read Data Cyclic Redundancy Checksum (**RDCRC**) in accordance with the IEEE 802.3 standard on DMA move data as it passes through the DMA.

#### Move Operation or Shadow Operation

To calculate a **DMA Data Checksum** for DMA move data the DMA channel RDCR must be initialized (e.g. written with  $00000000_H$  or another desired initial value) in addition to the standard DMA transaction configuration (source address, etc.) for the size of DMA move data.

If no error or retry event is reported during DMA move then the ME calculates an updated **DMA Data Checksum** value for each DMA move. On completion of the DMA transaction the ME stores the **DMA Data Checksum** in the DMA channel RDCR. Software may compare the calculated **DMA Data Checksum** with an expected **DMA Data Checksum** to verify the integrity of DMA move data.

#### Swap Byte

The **DMA Data Checksum** shall be calculated for all DMA channel data widths and may be configured to swap the byte order (at **Figure 220**). If DMA channel swap byte is selected (CHCFGR.SWAP =  $1_B$ ) then the order of bytes are swapped before **DMA Data Checksum** computation. Big-endian input is converted to little-endian and vice versa. If the DMA channel is configured for 8-bit channel data width (CHCFGR.CHDW =  $000_B$ ) then swap byte has no effect.

Swap byte does not change the byte order of data between a DMA read move and a DMA write move.

#### DMA Timestamp

If the DMA channel is configured to append a DMA timestamp then the DMA timestamp is not part of the **DMA Data Checksum** calculation.

#### Double Buffering Operations

The **DMA Data Checksum** is accumulated across all the DMA move data moved into both buffers. Software may re-initialize the **DMA Data Checksum** after a **Software Switch** and before the next data sample is received.

#### DMA Linked List (DMALL)

The **DMA Data Checksum** is initialized to  $00000000_H$  at the start of each DMA transaction.

#### Accumulated Linked List (ACLLL), Safe Linked List (SAFLL) and Conditional Linked List (CONLL)

The **DMA Data Checksum** is accumulated across all the DMA transactions in the linked list.

## Direct Memory Access (DMA)

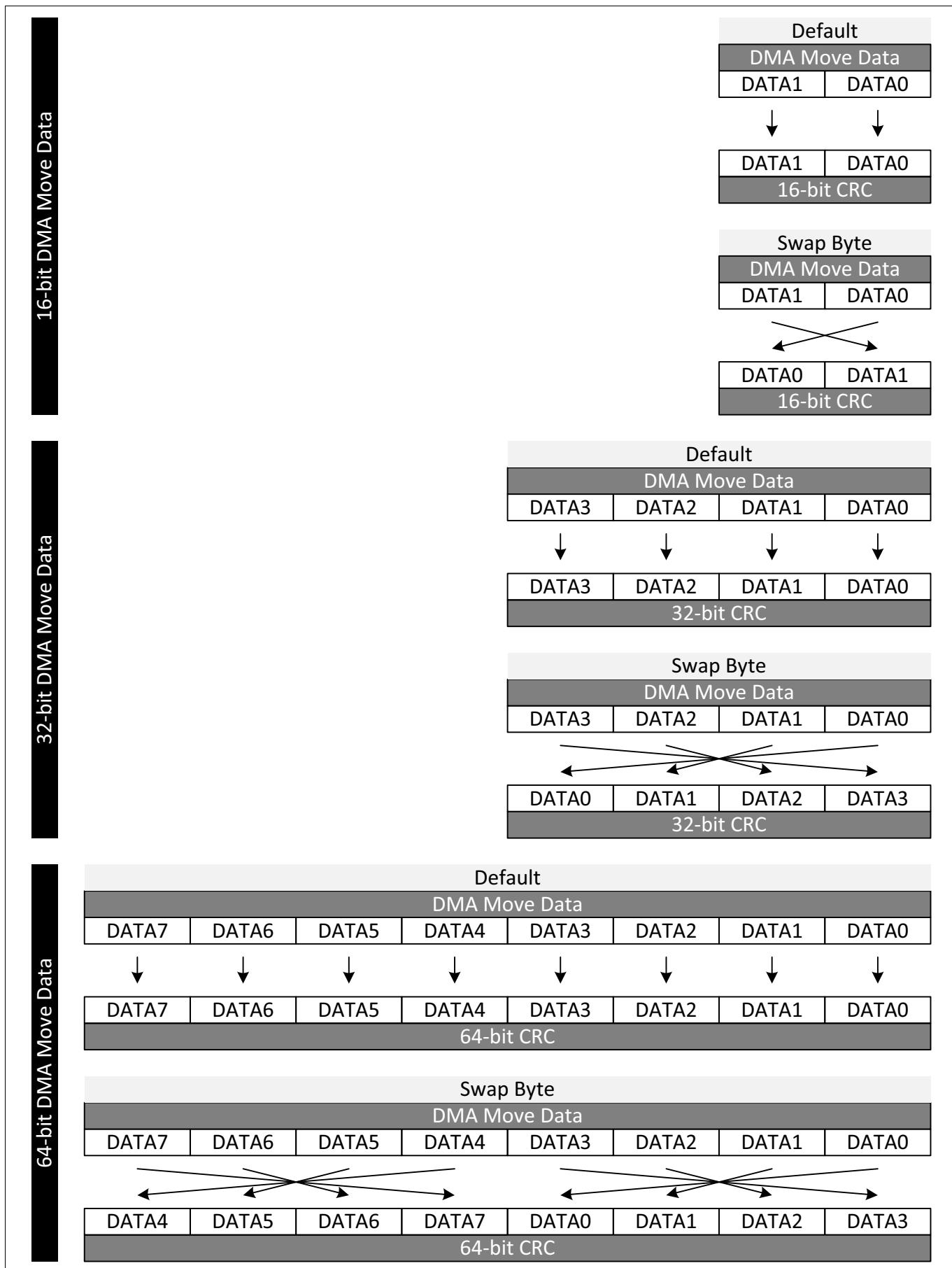


Figure 220 DMA Data Checksum - Swap Byte Order

## Direct Memory Access (DMA)

### 18.3.4.11 DMARAM Initialization

On the assertion of an application reset, the DMA shall initialize all DMARAM data to all zeros to prevent the triggering of DMARAM data integrity errors in the application.

### 18.3.5 Move Engine

Any ME shall service a DMA request from any DMA channel. As soon as a DMA channel wins arbitration, the TCS is copied from the DMARAM to the ME active channel registers and the DMA request is serviced. The ME requests the required buses and loads or stores data according to the TCS parameters of the active DMA channel. The ME is able to wait if the source or destination target is not available.

The processing of a DMA transfer (composed of several DMA moves) by the ME cannot be interrupted and is always completed. A DMA channel interrupt, reset, halt request or debug suspend only becomes active when the current DMA transfer is complete. Error conditions are reported. On completion of a DMA transaction or when a DMA channel loses channel arbitration the ME shall write the TCS back to the DMARAM.

#### 18.3.5.1 ME Read Buffer

The ME stores the DMA read move data in eight 32-bit read registers. If a DMA channel is configured for 8-bit, 16-bit, 32-bit, 64-bit or 128-bit channel data width and a DMA read move is to a cached address<sup>1)</sup>, the ME shall translate the read access to the on chip bus into a BTR4 access to a 32-byte aligned address. The ME stores the DMA read move data in the eight ME read registers together with the 32-byte aligned address to function as a ME read buffer as follows:

- **Hit**, if the 32-byte aligned address of the next DMA read move matches the 32-byte aligned address stored in the ME read buffer then the DMA read move data is read from the ME read buffer. No read access to the on chip bus occurs.
- **Miss**, if the 32-byte aligned address of the next DMA read move does not match the 32-byte aligned address stored in the ME read buffer then the ME invalidates the contents of the ME buffer. The ME executes the DMA read move by performing a read access to the on chip bus.

The ME invalidates the ME read buffer at the start of a DMA transfer.

##### 18.3.5.1.1 DMA Address Checksum

If a DMA read move is to a cached address, then the ME shall calculate the **DMA Address Checksum** using the 32-byte aligned address source address.

#### 18.3.5.2 ME Error Conditions

The ME reports TCS and source/destination error status. In the case of multiple errors, the error bits are set according to the error conditions (i.e. more than one error flag may be set). For all **ME Error Conditions**, the DMA shall perform the following actions:

- The ME shall record the number of the DMA channel in the ME last error channel bit field ERRSR.LEC.
- If the DMA channel is enabled for DMA channel hardware request then DMA channel TSR.HTRE is cleared.

#### DMARAM Integrity Error

When a DMA channel wins arbitration, a ME accesses the DMARAM to read the TCS. If an ECC error is detected, the ME shall set the error flag ERRSRm.RAMER. The DMA transaction shall not take place.

1) Segments 8 and 9 (see MEMMAP for details).

## Direct Memory Access (DMA)

### Source and Destination Errors

**SER** and **DER** include access protection errors and unsupported types of bus transaction.

- ME SER flag ERRSR.SER indicates an error occurred during a DMA read move from a source address.
  - ME DER error flag ERRSR.DER indicates an error occurred during a DMA write move to a destination address.
- If a SER or DER is reported then the ME completes the DMA transaction. If a SER is reported during a DMA read move then the DMA write move is not executed, but the destination address is updated.
- The ME bus error flag ERRSR.SPBER indicates an SPB bus error occurred during a DMA move.
  - The ME bus error flag ERRSR.SRIER indicates an SRI bus error occurred during a DMA move.

### Linked List Operation TCS Load Error

During DMALL, ACCL, SAFLL and CONLL operations if an error is reported during the loading of the next TCS from the on chip bus, the ME shall set the ME error flag ERRSR.DLLER. The linked list operation shall be aborted and the DMA channel transaction request state bit TSR.CH bit cleared.

### SAFLL DMA Address Checksum Error

The ME compares the **DMA Address Checksum** calculated by the current DMA transaction against the expected **DMA Address Checksum** stored in the next TCS. If the checksums do not match then the ME shall set the ME error flag ERRSR.SLLER.

### 18.3.5.3 Error Interrupt Service Request

#### 18.3.5.3.1 DMARAM Integrity Error Interrupt Service Request

A **RAMER** indicates that the ME detected a DMARAM integrity error:

- A RAMER is indicated by the ME error status flag ERRSRm.RAMER
- The DMA shall trigger a **DMA RP Error Interrupt Service Request**
- If software sets CLREm.CRAMER, the DMA shall clear error status flag ERRSRm.RAMER

#### 18.3.5.3.2 Source and Destination Error Interrupt Service Request

The ME shall detect the following error conditions (see [Figure 221](#)):

- A **SER** indicates a bus error occurred during a DMA read move from a source address.
  - A SER is indicated by the ME error status flag ERRSRm.SER
  - If EERm.ESER is set, the DMA shall trigger a **DMA RP Error Interrupt Service Request**.
  - If software sets CLREm.CSER, the DMA shall clear error status flag ERRSRm.SER
- A **DER** indicates a bus error that occurred during a DMA write move to a destination address.
  - A DER is indicated by the ME error status flag ERRSRm.DER
  - If EERm.EDER is set, the DMA shall trigger a **DMA RP Error Interrupt Service Request**.
  - If software sets CLREm.CDER, the DMA shall clear error status flag ERRSRm.DER

## Direct Memory Access (DMA)

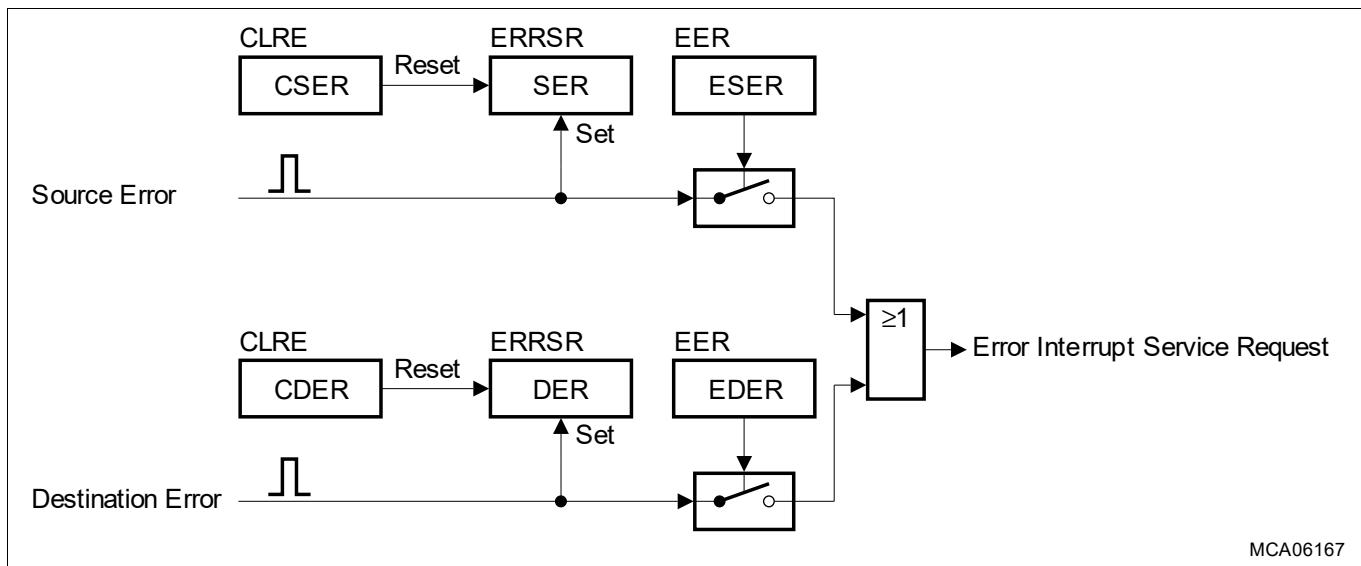


Figure 221 Source and Destination Error Interrupt Service Request

### 18.3.5.3.3 Linked List Operation TCS Error Interrupt Service Request

A **DLLER** indicates the ME detects an error when loading the next TCS from the on chip bus.

- A DLLER is indicated by the ME error status flag ERRSRm.DLLER
- The DMA shall trigger a **DMA RP Error Interrupt Service Request**.
- If software sets CLREm.CDLLER, the DMA shall clear error status flag ERRSRm.DLLER

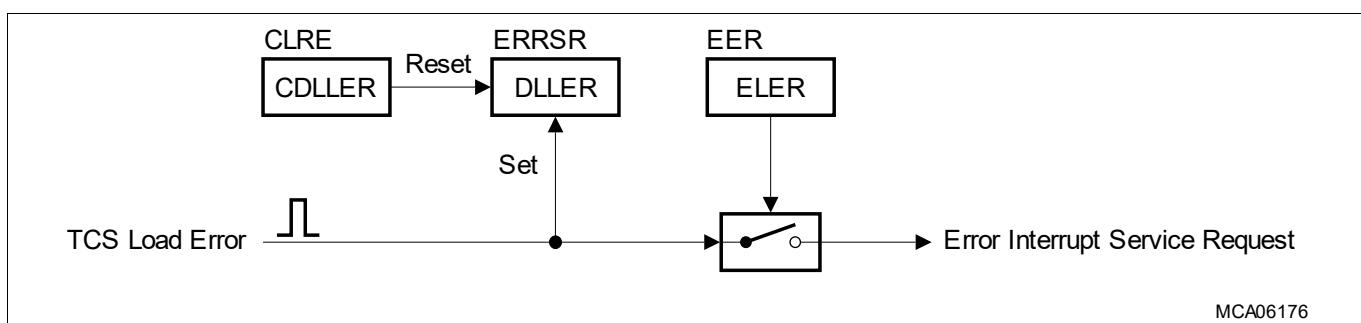


Figure 222 Linked List Operation TCS Error Interrupt Service Request

### 18.3.5.3.4 SAFLL DMA Address Checksum Error Interrupt Service Request

A **SLLER** indicates the ME detected a difference between the calculated and expected **DMA Address Checksum**.

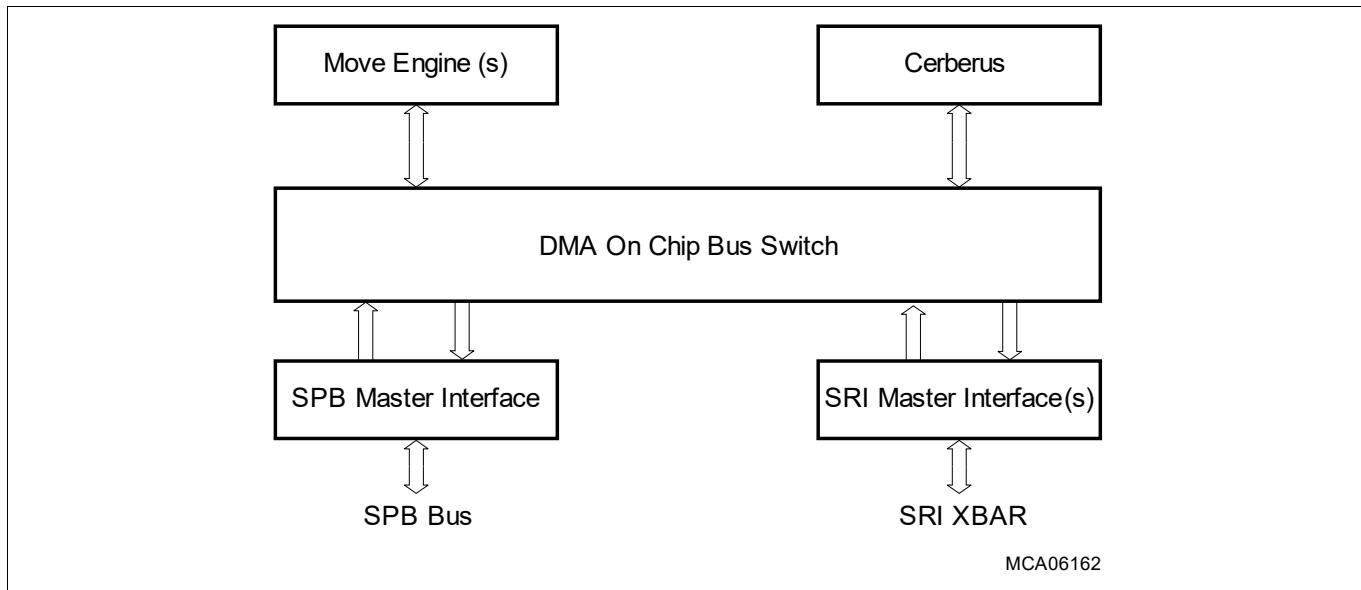
- A SLLER is indicated by the ME error status flag ERRSRm.SLLER
- The DMA shall trigger a **DMA RP Error Interrupt Service Request**.
- If software sets CLREm.CSLLER, the DMA shall clear error status flag ERRSRm.SLLER

## 18.3.6 DMA On Chip Bus

### 18.3.6.1 DMA On Chip Bus Switch

The DMA on chip bus switch (at [Figure 223](#)) arbitrates between each on chip bus request from the ME and Cerberus. Address routing across the whole memory map directs a request to an on chip bus master interface.

## Direct Memory Access (DMA)



**Figure 223 DMA On Chip Bus Switch**

### 18.3.6.1.1 SRI Master Interfaces

For production devices the DMA shall instance one SRI master interface (MIF0). For emulation devices the DMA may instance additional SRI master interfaces (MIF1 and MIF2)<sup>1)</sup> to be used as follows:

- DMA moves to EMEM Memory Banks 0 and 2 shall be routed through SRI master interface MIF1.
- DMA moves to EMEM Memory Banks 1 and 3 shall be routed through SRI master interface MIF2.

### 18.3.6.1.2 DMA On Chip Bus Switch Arbitration

If a ME and Cerberus request collide, the DMA on chip bus switch priorities (at **Table 580**) determine arbitration.

**Table 580 DMA On Chip Bus Switch Priorities**

Priority	Agent Requests	Comment
Highest	Cerberus to On Chip Bus High Priority.	Priority selection by software in Cerberus.
	ME write	For concurrent requests, the highest DMA channel wins.
	ME read	For concurrent requests, the highest DMA channel wins.
Lowest	Cerberus to On Chip Bus Low Priority.	Priority selection by software in Cerberus.

The on chip bus supports pipelining. High and low priority ME0, ME1 and Cerberus requests may be pipelined.

### 18.3.6.2 On Chip Bus Master Interfaces

The DMA instances on chip bus master interfaces to the SPB Bus and to the SRI Bus.

#### The DMA SPB master interface supports

- Single data read and write transactions (8-bit, 16-bit, 32-bit).
- Generation of interleaved FPI transactions from different access requesters (ME and Cerberus).
- De-assertion of request after retry in order to prevent bus blocking.

1) See MEMMAP and On-Chip System Connectivity for details.

## Direct Memory Access (DMA)

- Out of order transactions from different sources in order to avoid side effects (blocking) between the different access requesters (ME and Cerberus).

### The DMA SRI master interface supports

- Single data read and write transactions (8-bit, 16-bit, 32-bit, 64-bit)<sup>1)</sup>.
- Block transfer read and write transactions (128-bit, 256-bit)<sup>2)</sup>.
- Generation of interleaved SRI transactions from different access requesters (ME and Cerberus).

### Read Modify Write (RMW) Support

The DMA does not support RMW accesses to the on chip buses.

#### 18.3.6.3 SRI Alarm

The SRI bus protocol supports the reliable delivery of data between sources and destinations by extending the protocol to include ECC protection. During a DMA read move a ME checks the integrity of data received from SRI bus sources. A ME stores the DMA move data with ECC protection in the ME registers. During a DMA write move a ME generates ECC information compliant with the SRI bus protocol for write accesses to SRI destinations. A ME shall check the data integrity of DMA move data at the following locations:

- During a DMA read move the ME checks the integrity of data received from SRI source addresses.
- During a DMA write move the ME checks the integrity of data stored in the ME read register(s).

If a ME detects an ECC error then the DMA shall trigger an **SRI Alarm** to the SMU.

### System Peripheral Bus

DMA read move data received from an SPB source is not ECC protected. A ME shall generate ECC protection before the ME stores the data and ECC in the ME read register(s).

### Data Integrity Testing

Software may trigger an **SRI Alarm**. The CPU (CPUx\_SEGEN.ADFLIP = 01<sub>B</sub> and CPUx\_SEGEN.ADTYPE = 10<sub>B</sub>) may be used to generate an ECC error condition in a CPU slave interface used as a DMA read move source. When the data passes through the DMA it will generate an ECC error condition.

#### 18.3.7 Power Modes

The DMA disable status bit (DMA\_CLC.DISS) reports the state of the internal DMA clock signal ( $f_{DMA}$ ).

After the assertion of an application reset, the DMA is enabled (DMA\_CLC.DISS = 0<sub>B</sub>) i.e.  $f_{DMA}$  is enabled.

##### 18.3.7.1 Sleep Mode

The DMA shall enter **Sleep Mode** as follows:

- If the sleep mode enable control bit is enabled (DMA\_CLC.EDIS = 0<sub>B</sub>) AND the DMA sleep control is activated, the DMA shall enter **Sleep Mode** when the ME(s) have completed any pending DMA transfers.
- If software writes a DMA Disable Request (DMA\_CLC.DISR = 1<sub>B</sub>), the DMA shall enter **Sleep Mode** when the ME(s) have completed any pending DMA transfers.

If the DMA is in **Sleep Mode** (DMA\_CLC.DISS = 1<sub>B</sub>):

- The DMA shall disable  $f_{DMA}$  to minimize DMA power consumption.

1) 64-bit DMA move supported for DMA read move from SRI source to DMA write move to SRI destination.

2) Block transfer DMA move supported for DMA read move from SRI source to DMA write move to SRI destination.

## Direct Memory Access (DMA)

- DMA SFRs
    - If the DMA is in **Sleep Mode** AND software writes to DMA\_CLC, the DMA shall update DMA\_CLC.
    - If the DMA is in **Sleep Mode** AND software writes to other DMA SFRs, the DMA shall return a bus error.
    - If the DMA is in **Sleep Mode** AND software reads a DMA SFR, the DMA shall complete the read access.
  - DMARAM
    - If the DMA is in **Sleep Mode** AND software writes the DMARAM, the DMA shall return a bus error.
    - If the DMA is in **Sleep Mode** AND software reads the DMARAM, the DMA shall complete the read access.
- If the DMA sleep control is de-activated AND software clears the DMA disable request (DMA\_CLC.DISR = 0<sub>B</sub>):
- The DMA shall enable f<sub>DMA</sub>.
  - The DMA shall resume the servicing of DMA requests.

## Direct Memory Access (DMA)

### 18.4 Register

The DMA registers are accessed through the DMA slave interface.

**Table 581 Register Address Space - DMA**

Module	Base Address	End Address	Note
	F0010000 <sub>H</sub>	F0013FFF <sub>H</sub>	FPI slave interface

**Table 582 Register Overview - DMA (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
CLC	DMA Clock Control Register	0000 <sub>H</sub>	U,SV	SV,E,P00,P01	Application Reset	<a href="#">60</a>
ID	DMA Identification Register	0008 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">60</a>
ACCENr0	RP r Access Enable Register 0	0040 <sub>H</sub> +r*8	U,SV	SV,SE	Application Reset	<a href="#">64</a>
ACCENr1	RP r Access Enable Register 1	0044 <sub>H</sub> +r*8	U,SV	nBE	Application Reset	<a href="#">65</a>
EERm	ME m Enable Error Register	0120 <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	SV	Application Reset	<a href="#">80</a>
ERRSRm	ME m Error Status Register	0124 <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">80</a>
CLREm	ME m Clear Error Register	0128 <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	SV	Application Reset	<a href="#">82</a>
MEmSR	ME m Status Register	0130 <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">83</a>
MEm0R	ME m Read Register 0	0140 <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">84</a>
MEm1R	ME m Read Register 1	0144 <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">84</a>
MEm2R	ME m Read Register 2	0148 <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">85</a>
MEm3R	ME m Read Register 3	014C <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">85</a>
MEm4R	ME m Read Register 4	0150 <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">86</a>
MEm5R	ME m Read Register 5	0154 <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">86</a>
MEm6R	ME m Read Register 6	0158 <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">87</a>
MEm7R	ME m Read Register 7	015C <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">87</a>

## Direct Memory Access (DMA)

Table 582 Register Overview - DMA (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
MEmRDCRC	ME m Channel Read Data CRC Register	0180 <sub>H</sub> +m *1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">87</a>
MEmSDCRC	ME m Channel Source and Destination Address CRC Register	0184 <sub>H</sub> +m *1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">88</a>
MEmSADR	ME m Channel Source Address Register	0188 <sub>H</sub> +m *1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">88</a>
MEmDADR	ME m Channel Destination Address Register	018C <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">89</a>
MEmADICR	ME m Channel Address and Interrupt Control Register	0190 <sub>H</sub> +m *1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">89</a>
MEmCHCR	ME m Channel Control Register	0194 <sub>H</sub> +m *1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">91</a>
MEmSHADR	ME m Channel Shadow Address Register	0198 <sub>H</sub> +m *1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">91</a>
MEmCHSR	ME m Channel Status Register	019C <sub>H</sub> +m*1000 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">92</a>
OTSS	DMA OCDS Trigger Set Select	1200 <sub>H</sub>	U,SV	SV	See page <a href="#">61</a>	<a href="#">61</a>
PRR0	DMA Pattern Read Register 0	1208 <sub>H</sub>	U,SV	SV	Application Reset	<a href="#">62</a>
PRR1	DMA Pattern Read Register 1	120C <sub>H</sub>	U,SV	SV	Application Reset	<a href="#">62</a>
TIME	DMA Time Register	1210 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">63</a>
MODER	RP r Mode Register	1300 <sub>H</sub> +r*4	U,SV	SV,SE,P00, P01	Application Reset	<a href="#">63</a>
ERRINTRr	RP r Error Interrupt Set Register	1320 <sub>H</sub> +r*4	U,SV	SV,Pr	Application Reset	<a href="#">64</a>
HRRc	DMA Channel c Resource Partition Register	1800 <sub>H</sub> +c*4	U,SV	SV,SE,P00, P01	Application Reset	<a href="#">65</a>
SUSENRc	DMA Channel c Suspend Enable Register	1A00 <sub>H</sub> +c*4	U,SV	SV,E,Pr	See page <a href="#">65</a>	<a href="#">65</a>
SUSACRc	DMA Channel c Suspend Acknowledge Register	1C00 <sub>H</sub> +c*4	U,SV	BE	See page <a href="#">66</a>	<a href="#">66</a>
TSRc	DMA Channel c Transaction State Register	1E00 <sub>H</sub> +c*4	U,SV	SV,Pr	Application Reset	<a href="#">67</a>
RDCRCRc	DMARAM Channel c Read Data CRC Register	2000 <sub>H</sub> +c*20 <sub>H</sub>	U,SV	SV,Pr	Application Reset	<a href="#">69</a>

## Direct Memory Access (DMA)

**Table 582 Register Overview - DMA (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
SDCRCrc	DMARAM Channel c Source and Destination Address CRC Register	2004 <sub>H</sub> +c *20 <sub>H</sub>	U,SV	SV,Pr	Application Reset	<b>69</b>
SADRc	DMARAM Channel c Source Address Register	2008 <sub>H</sub> +c *20 <sub>H</sub>	U,SV	SV,Pr	Application Reset	<b>70</b>
DADRc	DMARAM Channel c Destination Address Register	200C <sub>H</sub> +c *20 <sub>H</sub>	U,SV	SV,Pr	Application Reset	<b>70</b>
ADICRc	DMARAM Channel c Address and Interrupt Control Register	2010 <sub>H</sub> +c *20 <sub>H</sub>	U,SV	SV,Pr	Application Reset	<b>71</b>
CHCFG Rc	DMARAM Channel c Configuration Register	2014 <sub>H</sub> +c *20 <sub>H</sub>	U,SV	SV,Pr	Application Reset	<b>75</b>
SHADRc	DMARAM Channel c Shadow Address Register	2018 <sub>H</sub> +c *20 <sub>H</sub>	U,SV	SV,Pr	Application Reset	<b>77</b>
CHCSRc	DMARAM Channel c Control and Status Register	201C <sub>H</sub> +c *20 <sub>H</sub>	U,SV	SV,Pr	Application Reset	<b>77</b>

### 18.4.1 Safety Flip-Flops

Safety flip-flops are special flip-flops that implement a hardware mechanism capable to detect single event effects that may lead to single event upsets (bit flip). The configuration and control registers that are implemented with safety flip-flops are:

- TSRc.CH

## Direct Memory Access (DMA)

### 18.4.2 Register

#### DMA Clock Control Register

The Clock Control Register controls the internal  $f_{\text{DMA}}$  clock signal and sleep mode in order to control the power consumption. After the application of a reset, the DMA is enabled.

CLC

DMA Clock Control Register																(0000 <sub>H</sub> )				
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16					
0																r				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	0				
r																rw	r	rh	rw	

Field	Bits	Type	Description
<b>DISR</b>	0	rw	<b>Module Disable Request Bit</b> Used for enable/disable control of the DMA $0_B$ DMA enable is requested. $1_B$ DMA disable is requested.
<b>DISS</b>	1	rh	<b>Module Disable Status Bit</b> Bit indicates the current status of the DMA $0_B$ DMA is enabled. $1_B$ DMA is disabled.
<b>EDIS</b>	3	rw	<b>Sleep Mode Enable Control</b> Used for DMA sleep mode control. $0_B$ Sleep control enabled. DMA may enter sleep mode. $1_B$ Sleep control disabled. DMA shall not enter sleep mode.
<b>0</b>	2, 31:4	r	<b>Reserved</b> Read as 0; must be written with 0.

#### DMA Identification Register

DMA Identification Register																(0008 <sub>H</sub> )			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	MOD_NUMBER			
r																			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	MOD_TYPE			
r																MOD_REV			

## Direct Memory Access (DMA)

Field	Bits	Type	Description
<b>MOD_REV</b>	7:0	r	<b>Module Revision Number</b> This bit field defines the module revision number.
<b>MOD_TYPE</b>	15:8	r	<b>Module Type</b> The bit field is set to C0 <sub>H</sub> which defines the module as a 32-bit module.
<b>MOD_NUMBE R</b>	31:16	r	<b>Module Number Value</b> This bit field defines a module identification number.

### DMA OCDS Trigger Set Select

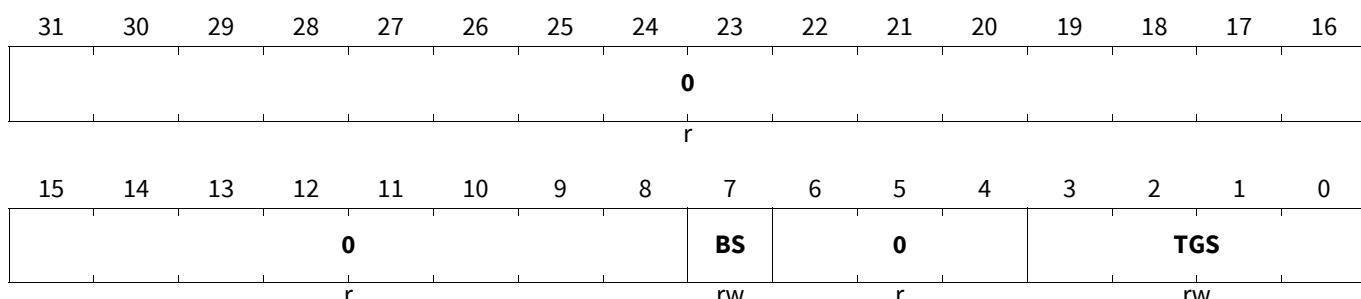
The OTSS register controls the selection of the OCDS Trigger Bus (OTGB).

The SCU controls the assertion of Power-on Reset to the OTSS register when OCDS is disabled or enabled. If OCDS is disabled then the OTSS register is reset by Power-on Reset else if OCDS is enabled then the OTSS register is not touched by Power-on Reset.

Write access requires OCDS to be enabled and Supervisor mode.

#### OTSS

**DMA OCDS Trigger Set Select** (1200<sub>H</sub>) Reset Value: [Table 583](#)



Field	Bits	Type	Description
<b>TGS</b>	3:0	rw	<b>Trigger Set for OTGB0 or OTGB1</b> 0 <sub>H</sub> No Trigger Set selected 1 <sub>H</sub> Trigger Set 1 2 <sub>H</sub> Trigger Set 2 3 <sub>H</sub> Reserved ... 7 <sub>H</sub> Reserved 8 <sub>H</sub> Trigger Set 8 9 <sub>H</sub> Trigger Set 9 A <sub>H</sub> Trigger Set 10 B <sub>H</sub> Trigger Set 11 C <sub>H</sub> Trigger Set 12 D <sub>H</sub> Trigger Set 13 E <sub>H</sub> Trigger Set 14 F <sub>H</sub> Trigger Set 15
<b>BS</b>	7	rw	<b>OTGB0 or OTGB1 Bus Select</b> 0 <sub>B</sub> Trigger Set is output on OTGB0 1 <sub>B</sub> Trigger Set is output on OTGB1

## Direct Memory Access (DMA)

Field	Bits	Type	Description
0	6:4, 31:8	r	<b>Reserved</b> Read as 0; must be written with 0.

**Table 583 Reset Values of OTSS**

Reset Type	Reset Value	Note
PowerOn Reset	0000 0000 <sub>H</sub>	
Debug Reset	0000 0000 <sub>H</sub>	

### DMA Pattern Read Register 0

The Pattern Read Register 0 stores the pattern data (mask and/or compare bits) to be processed by the ME pattern compare logic.

#### PRR0

DMA Pattern Read Register 0 <span style="float: right;">(1208<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub></span>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PAT03								PAT02							
rw								rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAT01								PAT00							
rw								rw							

Field	Bits	Type	Description
PAT00	7:0	rw	<b>Pattern Data Byte</b>
PAT01	15:8	rw	<b>Pattern Data Byte</b>
PAT02	23:16	rw	<b>Pattern Data Byte</b>
PAT03	31:24	rw	<b>Pattern Data Byte</b>

### DMA Pattern Read Register 1

The Pattern Read Register 1 stores the pattern data (mask and/or compare bits) to be processed by the ME pattern compare logic.

#### PRR1

DMA Pattern Read Register 1 <span style="float: right;">(120C<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub></span>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PAT13								PAT12							
rw								rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAT11								PAT10							
rw								rw							

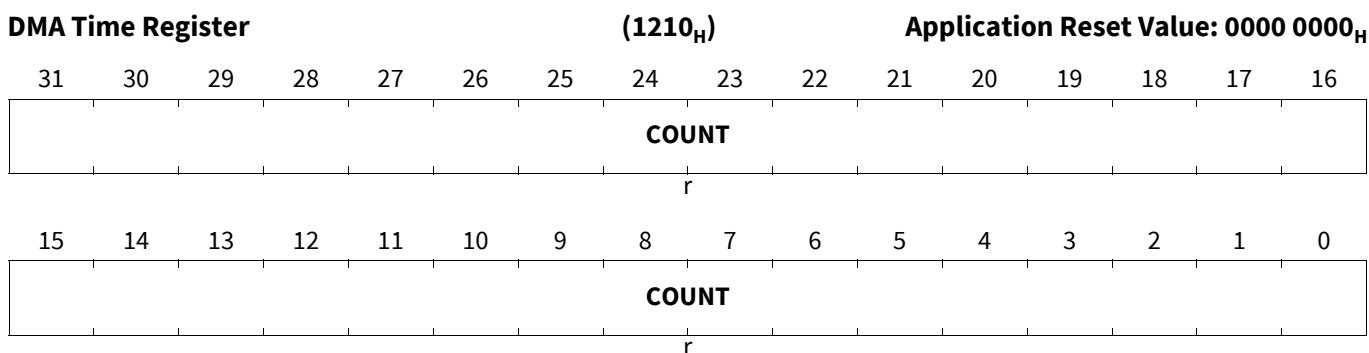
## Direct Memory Access (DMA)

Field	Bits	Type	Description
PAT10	7:0	rw	<b>Pattern Data Byte</b>
PAT11	15:8	rw	<b>Pattern Data Byte</b>
PAT12	23:16	rw	<b>Pattern Data Byte</b>
PAT13	31:24	rw	<b>Pattern Data Byte</b>

### DMA Time Register

The Time Register stores the 32-bit count value used for the appendage of DMA timestamps.

#### TIME



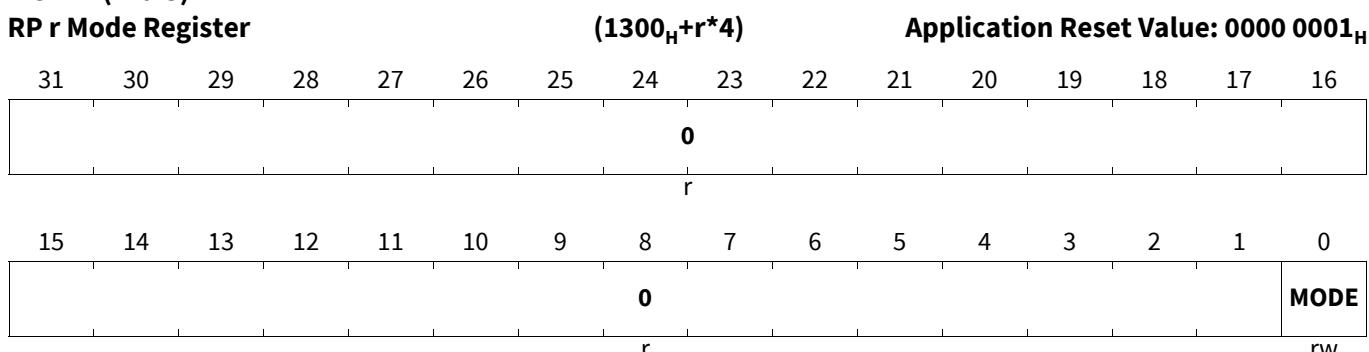
Field	Bits	Type	Description
COUNT	31:0	r	<b>Timestamp Count</b> The count value used during the appendage of DMA timestamps.

### 18.4.3 DMA Resource Partition Registers

#### RP r Mode Register

The Mode Register defines whether an RP accesses the on chip buses in supervisor mode or user mode.

#### MODEr (r=0-3)



Field	Bits	Type	Description
MODE	0	rw	<b>Resource Partition Supervisor Mode</b> 0 <sub>B</sub> Bus master interface accesses on chip bus in user mode. 1 <sub>B</sub> Bus master interface accesses on chip bus in supervisor mode.

**Direct Memory Access (DMA)**

Field	Bits	Type	Description
0	31:1	r	<b>Reserved</b> Read as 0; must be written with 0.

**RP r Error Interrupt Set Register****ERRINTR<sub>r</sub> (r=0-3)**

**RP r Error Interrupt Set Register** **(1320<sub>H</sub>+r\*4)** **Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															
SIT															
w															

Field	Bits	Type	Description
SIT	0	w	<b>Set Error Interrupt Service Request</b> Reading this bit returns a 0. $0_B$ No action. $1_B$ DMA error interrupt service request will be activated.
0	31:1	r	<b>Reserved</b> Read as 0; must be written with 0.

**RP r Access Enable Register 0****ACCENr0 (r=0-3)**

**RP r Access Enable Register 0** **(0040<sub>H</sub>+r\*8)** **Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

Field	Bits	Type	Description
ENq (q=0-31)	q	rw	<b>Access Enable for Master TAG ID q</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID q $0_B$ Write access will not be executed $1_B$ Write access will be executed

## Direct Memory Access (DMA)

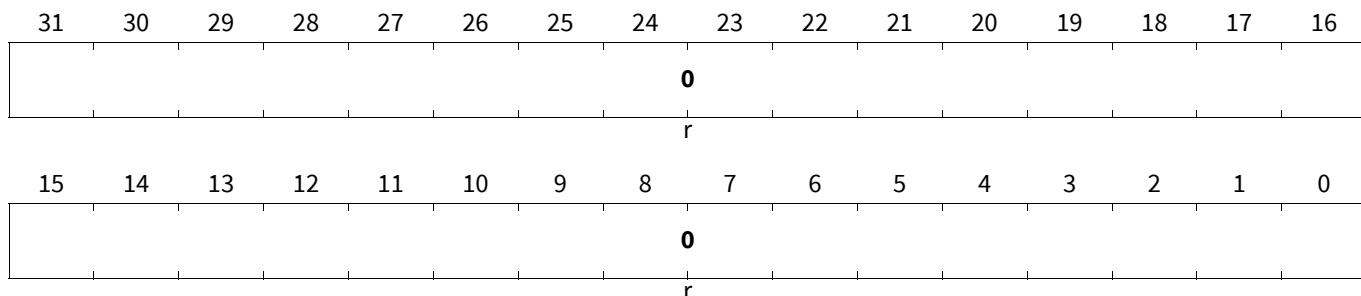
### RP r Access Enable Register 1

#### ACCENr1 (r=0-3)

##### RP r Access Enable Register 1

(0044<sub>H</sub>+r\*8)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
0	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

### 18.4.4 DMA Channel Registers

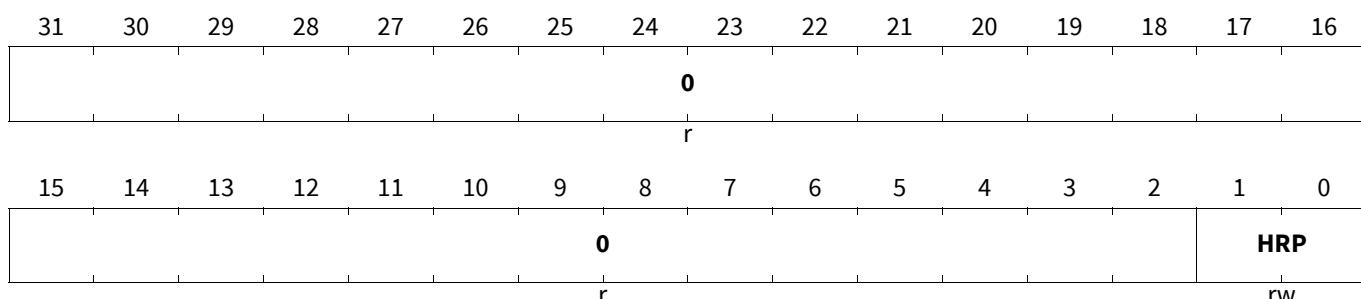
#### DMA Channel c Resource Partition Register

The Resource Partition Register assigns a DMA channel to a Resource Partition.

#### HRRc (c=000-127)

##### DMA Channel c Resource Partition Register (1800<sub>H</sub>+c\*4)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
HRP	1:0	rw	<b>DMA Channel Resource Partition</b> 00 <sub>B</sub> Resource Partition 0 (RP0). 01 <sub>B</sub> Resource Partition 1 (RP1). 10 <sub>B</sub> Resource Partition 2 (RP2). 11 <sub>B</sub> Resource Partition 3 (RP3).
0	31:2	r	<b>Reserved</b> Read as 0; must be written with 0.

#### DMA Channel c Suspend Enable Register

The Suspend Enable Register enables/disables soft suspend mode capability.

## Direct Memory Access (DMA)

### SUSENRc (c=000-127)

#### DMA Channel c Suspend Enable Register (1A00<sub>H</sub>+c\*4)

Reset Value: [Table 584](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
															0
									r						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															SUSEN
								0							rw
								r							

Field	Bits	Type	Description
<b>SUSEN</b>	0	rw	<b>Channel Suspend Enable for DMA Channel</b> Enables the DMA channel suspend capability. Channel suspend mode shall be terminated when SUSEN is written with 0. 0 <sub>B</sub> DMA channel is disabled for DMA channel suspend. The DMA channel does not react on an active suspend request signal SUSREQ. 1 <sub>B</sub> DMA channel is enabled for DMA channel suspend. If the suspend request signal SUSREQ becomes active, a DMA transaction of the DMA channel is stopped after the current DMA transfer has completed.
<b>0</b>	31:1	r	<b>Reserved</b> Read as 0; must be written with 0.

**Table 584 Reset Values of SUSENRc (c=000-127)**

Reset Type	Reset Value	Note
PowerOn Reset	0000 0000 <sub>H</sub>	
Debug Reset	0000 0000 <sub>H</sub>	

### DMA Channel c Suspend Acknowledge Register

The Suspend Acknowledge Register indicates the DMA Channel soft suspend status.

### SUSACRc (c=000-127)

#### DMA Channel c Suspend Acknowledge Register(1C00<sub>H</sub>+c\*4)

Reset Value: [Table 585](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
															0
									r						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															SUSAC
								0							rh
								r							

## Direct Memory Access (DMA)

Field	Bits	Type	Description
SUSAC	0	rh	<b>DMA Channel Suspend State or Frozen State Active for DMA Channel</b> Status bit indicates whether or not a DMA channel is in channel suspend state or in the frozen state.  $0_B$ DMA channel is not in channel suspend state, frozen state or internal actions are not completed after the channel suspend state or frozen state was requested. $1_B$ DMA channel is in channel suspend state or frozen state.
0	31:1	r	<b>Reserved</b> Read as 0; must be written with 0.

Table 585 Reset Values of SUSACRc (c=000-127)

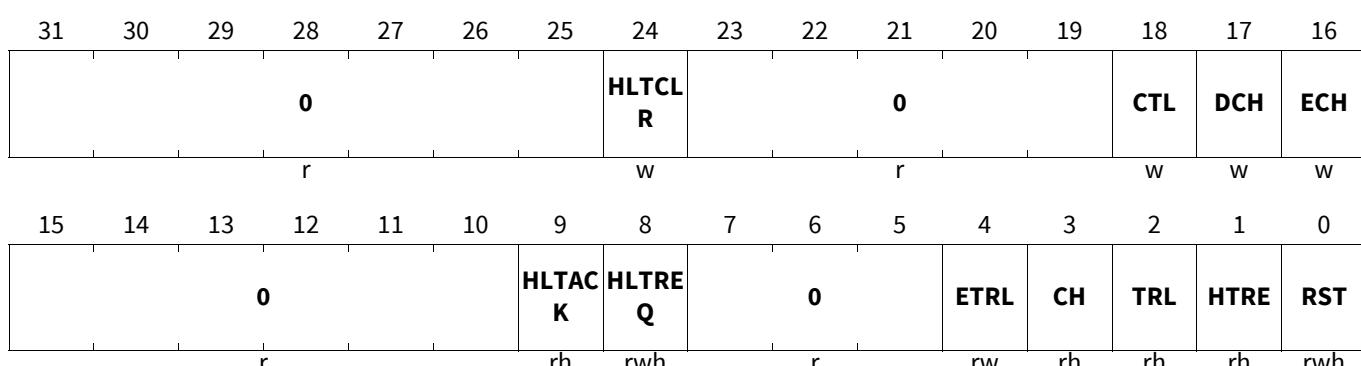
Reset Type	Reset Value	Note
PowerOn Reset	0000 0000 <sub>H</sub>	
Debug Reset	0000 0000 <sub>H</sub>	

## DMA Channel c Transaction State Register

If an application reset is asserted, the DMA shall set DMA channel reset to all DMA channels. As soon as a DMA channel has entered the reset state, the DMA shall clear the DMA channel bit.

## TSRc (c=000-127)

**DMA Channel c Transaction State Register (1E00<sub>H</sub>+c\*4) Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
RST	0	rwh	<b>DMA Channel Reset</b> The DMA channel reset bit is set by software (DMA channel TSR.RST = 1) and cleared by hardware when the DMA channel has been reset.  $0_B$ After the application of a DMA channel reset, the DMA channel is in the DMA channel reset state. Software write to 0 has no effect. $1_B$ A DMA channel reset is pending. See Functional Description.

**Direct Memory Access (DMA)**

Field	Bits	Type	Description
<b>HTRE</b>	1	rh	<p><b>DMA Channel Hardware Request Enable</b></p> <p>Note: See Functional Description for enable and disable. When a DMA channel is configured for single mode, HTRE is reset when ME CHSR.TCOUNT is decremented and ME CHSR.TCOUNT = 0. When a DMA channel error is reported or a pattern match is detected, DMA channel HTRE is reset.</p> <p>0<sub>B</sub> DMA hardware request is disabled for DMA channel. 1<sub>B</sub> DMA hardware request is enabled for DMA channel.</p>
<b>TRL</b>	2	rh	<p><b>DMA Channel Transaction/Transfer Request Lost</b></p> <p>This bit is reset by software clearing TRL (writing DMA channel TSR.CTL = 1) or resetting the DMA channel (writing DMA channel TSR.RST = 1).</p> <p>0<sub>B</sub> No TRL event has been detected for DMA channel. 1<sub>B</sub> TRL event has been detected for DMA channel.</p>
<b>CH</b>	3	rh	<p><b>DMA Channel Transaction Request State</b></p> <p>CH is reset when a pattern match is detected.</p> <p>0<sub>B</sub> No DMA request is pending for DMA channel. 1<sub>B</sub> DMA request is pending for DMA channel.</p>
<b>ETRL</b>	4	rw	<p><b>Enable DMA Channel Transaction/Transfer Request Lost Interrupt</b></p> <p>DMA channel control bit to enable the generation of an error interrupt service request when DMA channel TSR.TRL is set.</p> <p>0<sub>B</sub> Interrupt generation for DMA channel TRL event is disabled. 1<sub>B</sub> Interrupt generation for DMA channel TRL event is enabled.</p>
<b>HLTREQ</b>	8	rwh	<p><b>DMA Channel Halt Request</b></p> <p>The DMA channel halt request bit is set by software (writing DMA channel TSR.HLTREQ = 1) and cleared by software (writing DMA channel TSR.HLTCLR = 1).</p> <p>0<sub>B</sub> No action. 1<sub>B</sub> Halt request.</p>
<b>HLTACK</b>	9	rh	<p><b>DMA Channel Halt Acknowledge</b></p> <p>0<sub>B</sub> DMA channel is not halted. 1<sub>B</sub> DMA channel is halted.</p>
<b>ECH</b>	16	w	<p><b>Enable DMA Channel Hardware Transaction Request</b></p> <p>See Functional Description. Reading this bit returns a 0.</p>
<b>DCH</b>	17	w	<p><b>Disable DMA Channel Hardware Transaction Request</b></p> <p>See Functional Description. Reading this bit returns a 0.</p>
<b>CTL</b>	18	w	<p><b>Clear DMA Channel Transaction/Transfer Request Lost</b></p> <p>Software clear of the DMA channel TRL status flag.</p> <p>Reading this bit returns a 0.</p> <p>0<sub>B</sub> No action. 1<sub>B</sub> Clear DMA channel TRL flag (TSR.TRL).</p>

## Direct Memory Access (DMA)

Field	Bits	Type	Description
HLTCLR	24	w	<b>Clear DMA Channel Halt Request and Acknowledge</b> Reading this bit returns a 0. 0 <sub>B</sub> No action. 1 <sub>B</sub> Clear DMA channel halt request (TSR.HLTREQ) and halt acknowledge (TSR.HLTACK).
0	7:5, 15:10, 23:19, 31:25	r	<b>Reserved</b> Read as 0; must be written with 0.

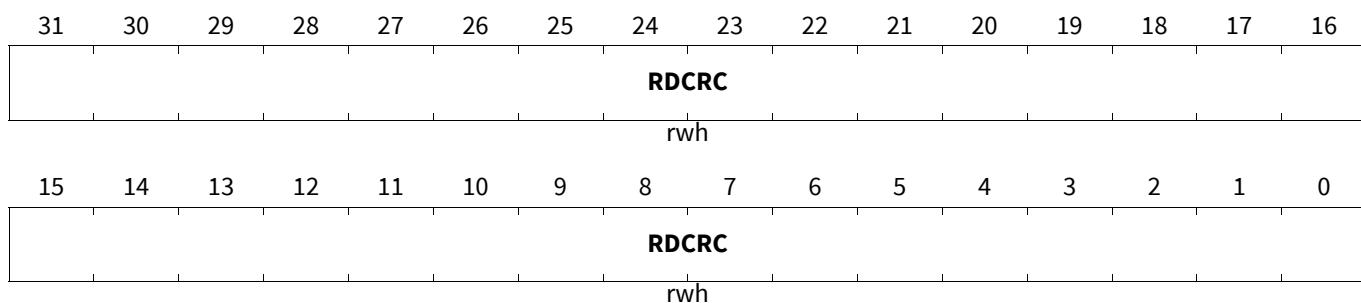
### 18.4.5 DMARAM Channel Registers

#### DMARAM Channel c Read Data CRC Register

RDCRCR<sub>c</sub> (c=000-127)

DMARAM Channel c Read Data CRC Register ( $2000_H + c * 20_H$ )

Application Reset Value: 0000 0000<sub>H</sub>

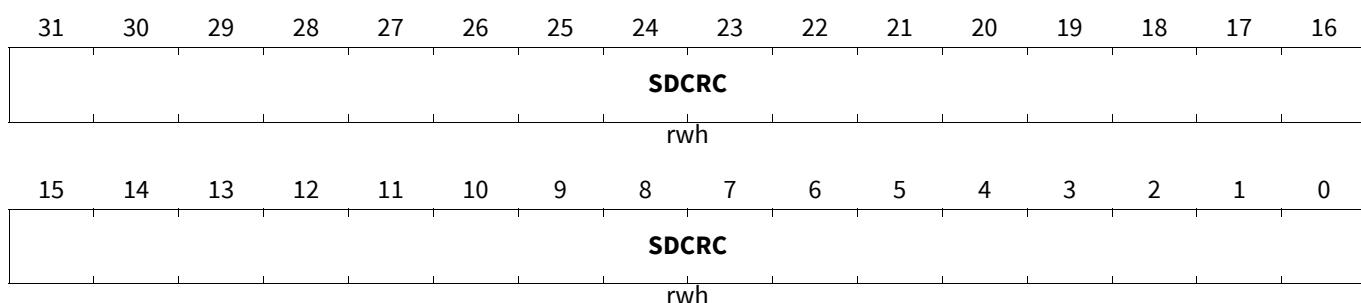


Field	Bits	Type	Description
RDCRC	31:0	rwh	<b>Read Data CRC</b> Checksum calculated for DMA read move data.

#### DMARAM Channel c Source and Destination Address CRC Register

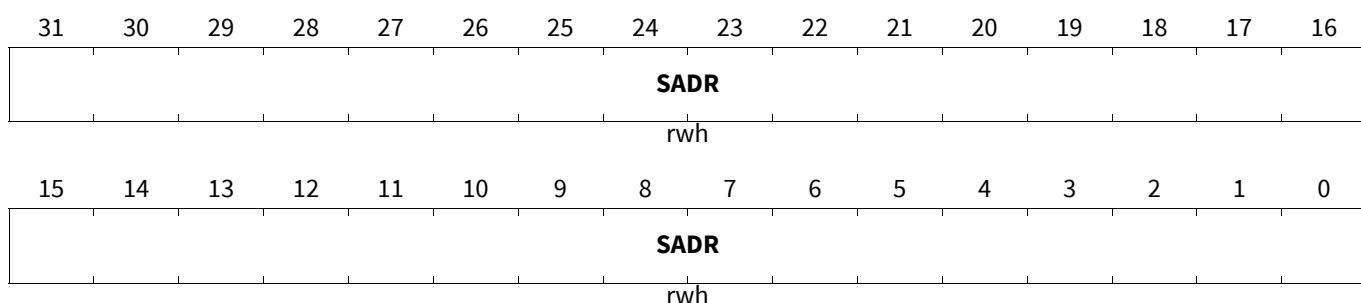
SDCRCR<sub>c</sub> (c=000-127)

DMARAM Channel c Source and Destination Address CRC Register( $2004_H + c * 20_H$ ) Application Reset Value:  
0000 0000<sub>H</sub>

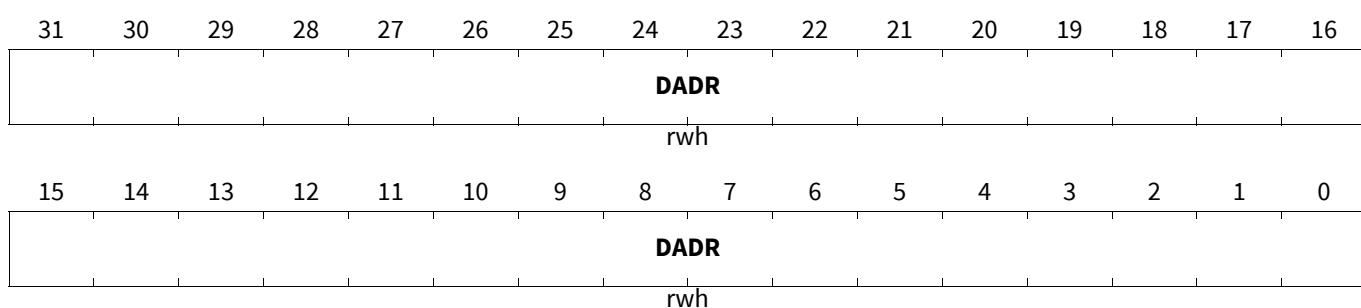


**Direct Memory Access (DMA)**

Field	Bits	Type	Description
<b>SDCRC</b>	31:0	rwh	<b>Source and Destination Address CRC</b> Checksum calculated for DMA move source and destination addresses.

**DMARAM Channel c Source Address Register****SADRC (c=000-127)****DMARAM Channel c Source Address Register ( $2008_H + c * 20_H$ )****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>SADR</b>	31:0	rwh	<b>Source Address</b> 32-bit source address.

**DMARAM Channel c Destination Address Register****DADRC (c=000-127)****DMARAM Channel c Destination Address Register( $200C_H + c * 20_H$ )****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>DADR</b>	31:0	rwh	<b>Destination Address</b> 32-bit destination address.

**Direct Memory Access (DMA)****DMARAM Channel c Address and Interrupt Control Register****ADICRc (c=000-127)**
**DMARAM Channel c Address and Interrupt Control Register(2010<sub>H</sub>+c\*20<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>IRDV</b>			<b>INTCT</b>		<b>WRPD E</b>	<b>WRPS E</b>	<b>0</b>	<b>STAM P</b>	<b>DCBE</b>	<b>SCBE</b>	<b>SHCT</b>				
rwh			rwh	rwh	rwh	rwh	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>CBLD</b>			<b>CBLS</b>			<b>INCD</b>	<b>DMF</b>			<b>INCS</b>	<b>SMF</b>				
rw			rw	rw		rwh		rwh	rwh	rwh	rwh	rw	rwh	rwh	

Field	Bits	Type	Description
<b>SMF</b>	2:0	rwh	<p><b>Source Address Modification Factor</b></p> <p>DMA channel TCS 32-bit source address modification factor and the channel data width CHDW determines an address offset value by which the source address is modified after each DMA move. If SCBE = 1<sub>B</sub> and CBLS = 0000<sub>B</sub> then the source address is not modified.</p> <p>000<sub>B</sub> Address offset is 1 x CHDW            001<sub>B</sub> Address offset is 2 x CHDW            010<sub>B</sub> Address offset is 4 x CHDW            011<sub>B</sub> Address offset is 8 x CHDW            100<sub>B</sub> Address offset is 16 x CHDW            101<sub>B</sub> Address offset is 32 x CHDW            110<sub>B</sub> Address offset is 64 x CHDW            111<sub>B</sub> Address offset is 128 x CHDW</p>
<b>INCS</b>	3	rwh	<p><b>Increment of Source Address</b></p> <p>DMA channel TCS control bit to determine if the address offset selected by SMF will be added to or subtracted from the source address after each DMA move. If SCBE = 1<sub>B</sub> and CBLS = 0000<sub>B</sub> then the source address is not modified.</p> <p>0<sub>B</sub> Address offset will be subtracted.            1<sub>B</sub> Address offset will be added.</p>

**Direct Memory Access (DMA)**

Field	Bits	Type	Description
<b>DMF</b>	6:4	rwh	<p><b>Destination Address Modification Factor</b></p> <p>DMA channel TCS 32-bit destination address modification factor and the channel data width CHDW determines an address offset value by which the destination address is modified after each DMA move. If DCBE = <math>1_B</math> and CBLD = <math>0000_B</math> then the destination address is not modified.</p> <ul style="list-style-type: none"> <li><math>000_B</math> Address offset is <math>1 \times</math> CHDW</li> <li><math>001_B</math> Address offset is <math>2 \times</math> CHDW</li> <li><math>010_B</math> Address offset is <math>4 \times</math> CHDW</li> <li><math>011_B</math> Address offset is <math>8 \times</math> CHDW</li> <li><math>100_B</math> Address offset is <math>16 \times</math> CHDW</li> <li><math>101_B</math> Address offset is <math>32 \times</math> CHDW</li> <li><math>110_B</math> Address offset is <math>64 \times</math> CHDW</li> <li><math>111_B</math> Address offset is <math>128 \times</math> CHDW</li> </ul>
<b>INCD</b>	7	rwh	<p><b>Increment of Destination Address</b></p> <p>DMA channel TCS control bit to determine if the address offset selected by DMF will be added to or subtracted from the destination address after each DMA move. If DCBE = <math>1_B</math> and CBLD = <math>0000_B</math> the destination address is not modified.</p> <ul style="list-style-type: none"> <li><math>0_B</math> Address offset will be subtracted.</li> <li><math>1_B</math> Address offset will be added.</li> </ul>
<b>CBLS</b>	11:8	rw	<p><b>Circular Buffer Length Source</b></p> <p>DMA channel TCS circular buffer source address update control bit determines which part of the 32-bit source address register remains unchanged and is not updated after a DMA move operation.</p> <p><i>Note:</i> CBLS determines the size of the circular source buffer.</p> <ul style="list-style-type: none"> <li><math>0_H</math> Source address SADR[31:0] is not updated</li> <li><math>1_H</math> Source address SADR[31:1] is not updated</li> <li><math>2_H</math> Source address SADR[31:2] is not updated</li> <li><math>3_H</math> Source address SADR[31:3] is not updated</li> <li><math>4_H</math> Source address SADR[31:4] is not updated</li> <li><math>5_H</math> Source address SADR[31:5] is not updated</li> <li><math>6_H</math> Source address SADR[31:6] is not updated</li> <li><math>7_H</math> Source address SADR[31:7] is not updated</li> <li><math>8_H</math> Source address SADR[31:8] is not updated</li> <li><math>9_H</math> Source address SADR[31:9] is not updated</li> <li><math>A_H</math> Source address SADR[31:10] is not updated</li> <li><math>B_H</math> Source address SADR[31:11] is not updated</li> <li><math>C_H</math> Source address SADR[31:12] is not updated</li> <li><math>D_H</math> Source address SADR[31:13] is not updated</li> <li><math>E_H</math> Source address SADR[31:14] is not updated</li> <li><math>F_H</math> Source address SADR[31:15] is not updated</li> </ul>

**Direct Memory Access (DMA)**

Field	Bits	Type	Description
<b>CBLD</b>	15:12	rw	<p><b>Circular Buffer Length Destination</b>            DMA channel TCS circular buffer destination address update control bit determines which part of the 32-bit destination address register remains unchanged and is not updated after a DMA move operation.</p> <p><i>Note:</i> CBLD determines the size of the circular destination buffer.</p> <p>0<sub>H</sub> Destination address DADR[31:0] is not updated            1<sub>H</sub> Destination address DADR[31:1] is not updated            2<sub>H</sub> Destination address DADR[31:2] is not updated            3<sub>H</sub> Destination address DADR[31:3] is not updated            4<sub>H</sub> Destination address DADR[31:4] is not updated            5<sub>H</sub> Destination address DADR[31:5] is not updated            6<sub>H</sub> Destination address DADR[31:6] is not updated            7<sub>H</sub> Destination address DADR[31:7] is not updated            8<sub>H</sub> Destination address DADR[31:8] is not updated            9<sub>H</sub> Destination address DADR[31:9] is not updated            A<sub>H</sub> Destination address DADR[31:10] is not updated            B<sub>H</sub> Destination address DADR[31:11] is not updated            C<sub>H</sub> Destination address DADR[31:12] is not updated            D<sub>H</sub> Destination address DADR[31:13] is not updated            E<sub>H</sub> Destination address DADR[31:14] is not updated            F<sub>H</sub> Destination address DADR[31:15] is not updated</p>
<b>SHCT</b>	19:16	rwh	<p><b>Shadow Control</b>            DMA channel TCS shadow control determines the function of the shadow address register.</p> <p>0<sub>H</sub> Move Operation.            1<sub>H</sub> Shadow Operation Read Only Mode Source Address.            2<sub>H</sub> Shadow Operation Read Only Mode Destination Address.            3<sub>H</sub> Reserved.            4<sub>H</sub> Reserved.            5<sub>H</sub> Shadow Operation Direct Write Mode Source Address.            6<sub>H</sub> Shadow Operation Direct Write Mode Destination Address.            7<sub>H</sub> Reserved.            8<sub>H</sub> DMA Double Source Buffering with Software Switch Only.            9<sub>H</sub> DMA Double Source Buffering with Software Switch and Automatic Hardware Switch.            A<sub>H</sub> DMA Double Destination Buffering with Software Switch Only.            B<sub>H</sub> DMA Double Destination Buffering with Software Switch and Automatic Hardware Switch.            C<sub>H</sub> DMA Linked List (DMALL).            D<sub>H</sub> Accumulated Linked List (ACLLL).            E<sub>H</sub> Safe Linked List (SAFLL).            F<sub>H</sub> Conditional Linked List (CONLL).</p>
<b>SCBE</b>	20	rwh	<p><b>Source Circular Buffer Enable</b>            DMA channel TCS source circular buffer enable.</p> <p>0<sub>B</sub> Source circular buffer disabled.            1<sub>B</sub> Source circular buffer enabled.</p>

**Direct Memory Access (DMA)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>DCBE</b>	21	rwh	<b>Destination Circular Buffer Enable</b> DMA channel TCS destination circular buffer enable. $0_B$ Destination circular buffer disabled. $1_B$ Destination circular buffer enabled.
<b>STAMP</b>	22	rwh	<b>Time Stamp</b> DMA channel TCS control bit to enable the appendage of a timestamp after the end of the last DMA Move during a DMA transaction. $0_B$ No action. $1_B$ DMA timestamp is appended.
<b>WRPSE</b>	24	rwh	<b>Wrap Source Enable</b> DMA channel TCS source buffer interrupt trigger enable/disable. $0_B$ Wrap source buffer interrupt trigger disabled. $1_B$ Wrap source buffer interrupt trigger enabled.
<b>WRPDE</b>	25	rwh	<b>Wrap Destination Enable</b> DMA channel TCS destination buffer interrupt trigger enable/disable. $0_B$ Wrap destination buffer interrupt trigger disabled. $1_B$ Wrap destination buffer interrupt trigger enabled.
<b>INTCT</b>	27:26	rwh	<b>Interrupt Control</b> DMA channel TCS interrupt control.  <i>Note:</i> If DMA channel CHCFGR.PRSEL = $1_B$ for the next lower priority channel then the channel transfer trigger interrupt is disabled.  $00_B$ No interrupt trigger will be generated on changing the TCOUNT value. The DMA channel CHSR.ICH is set when TCOUNT equals IRDV. $01_B$ No interrupt trigger will be generated on changing the TCOUNT value. The DMA channel CHSR.ICH is set when TCOUNT is decremented $10_B$ Interrupt trigger is generated and DMA channel CHSR.ICH is set on changing the TCOUNT value and TCOUNT equals IRDV $11_B$ Interrupt trigger is generated and DMA channel CHSR.ICH is set each time TCOUNT is decremented
<b>IRDV</b>	31:28	rwh	<b>Interrupt Raise Detect Value</b> DMA channel TCS interrupt threshold value defines the Threshold Limit of CHSR.TCOUNT for which a channel interrupt trigger will be raised.
<b>0</b>	23	r	<b>Reserved</b> Read as 0; must be written with 0.

**Direct Memory Access (DMA)****DMARAM Channel c Configuration Register****CHCFG Rc (c=000-127)****DMARAM Channel c Configuration Register (2014<sub>H</sub>+c\*20<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		0	PRSEL	SWAP		PATSEL			CHDW		CHMO DE	RROA T		BLKM	
r		rwh	rw		rwh			rwh	rwh	rwh	rwh	rwh	rwh	rwh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0								TREL						
r									rwh						

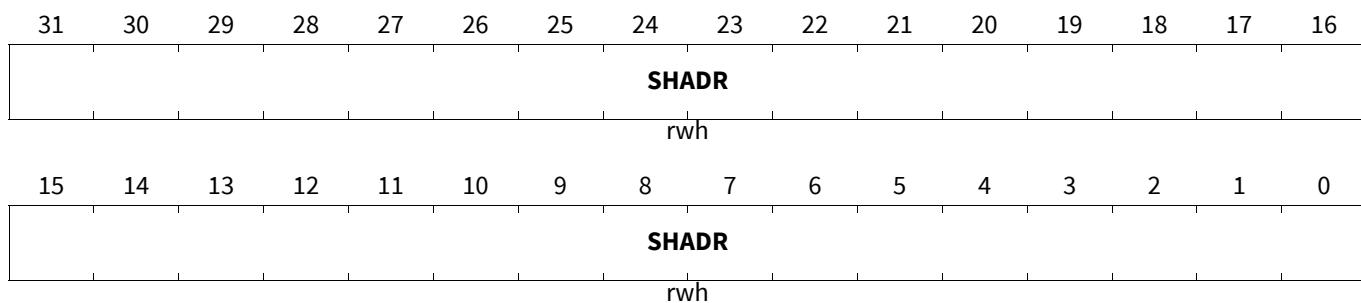
Field	Bits	Type	Description
<b>TREL</b>	13:0	rwh	<p><b>Transfer Reload Value</b>            DMA channel TCS transfer reload value to control the number of DMA transfers in a DMA transaction. The 14-bit transfer count value is loaded into ME CHSR.TCOUNT at the start of a DMA transaction (when TSR.CH becomes set and CHSR.TCOUNT = 0). A write to CHCFG.R.TREL during a running DMA transaction has no influence on the running DMA transaction.            If CHCFG.R.TREL = 0 or if CHCFG.R.TREL = 1 then ME CHSR.TCOUNT will be loaded with 1 when a new DMA transaction is started (at least one DMA transfer must be executed per DMA transaction).</p>
<b>BLKM</b>	18:16	rwh	<p><b>Block Mode</b>            Defines the number of DMA moves executed during one DMA transfer.</p> <ul style="list-style-type: none"> <li>000<sub>B</sub> One DMA transfer has 1 DMA move</li> <li>001<sub>B</sub> One DMA transfer has 2 DMA moves</li> <li>010<sub>B</sub> One DMA transfer has 4 DMA moves</li> <li>011<sub>B</sub> One DMA transfer has 8 DMA moves</li> <li>100<sub>B</sub> One DMA transfer has 16 DMA moves</li> <li>101<sub>B</sub> One DMA transfer has 3 DMA moves</li> <li>110<sub>B</sub> One DMA transfer has 5 DMA moves</li> <li>111<sub>B</sub> One DMA transfer has 9 DMA moves</li> </ul>
<b>RROAT</b>	19	rwh	<p><b>Reset Request Only After Transaction</b>            DMA channel control bit to determine if the DMA request state flag (DMA channel TSR.CH) is reset after each DMA transfer.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> DMA channel TSR.CH is reset after the start of each DMA transfer. A DMA request is required for each DMA transfer.</li> <li>1<sub>B</sub> DMA channel TSR.CH is reset when CHSR.TCOUNT = 0 and after the completion of the last DMA transfer (i.e. on completion of the DMA transaction). One DMA request starts a complete DMA transaction.</li> </ul>

## Direct Memory Access (DMA)

Field	Bits	Type	Description
<b>CHMODE</b>	20	rwh	<p><b>Channel Operation Mode</b></p> <p>DMA channel TCS control to determine TSR.HTRE reset condition.</p> <p><math>0_B</math> <b>Single_Mode</b>, is selected for DMA channel. After a DMA transaction, DMA channel is disabled for further hardware requests (TSR.HTRE is reset by hardware) TSR.HTRE must be set again by software for starting a new transaction.</p> <p><math>1_B</math> <b>Continuous_Mode</b>, is selected for DMA channel. After a DMA transaction, bit TSR.HTRE remains set.</p>
<b>CHDW</b>	23:21	rwh	<p><b>Channel Data Width</b></p> <p>DMA channel TCS data width for DMA read moves and DMA write moves.</p> <p><math>000_B</math> 8-bit data width for moves selected  <i>Note: Single Data Transfer Byte (SDTB)</i></p> <p><math>001_B</math> 16-bit data width for moves selected  <i>Note: Single Data Transfer Half-Word (SDTH)</i></p> <p><math>010_B</math> 32-bit data width for moves selected  <i>Note: Single Data Transfer Word (SDTW)</i></p> <p><math>011_B</math> 64-bit data width transaction selected  <i>Note: SRI-Bus: Single Data Transfer Double Word (SDTD)</i>  <i>Note: SPB-Bus: not supported.</i></p> <p><math>100_B</math> 128-bit data width transaction selected  <i>Note: SRI-Bus: Block Transfer Request - 2 Transfers (BTR2)</i>  <i>Note: SPB-Bus: not supported.</i></p> <p><math>101_B</math> 256-bit data width transaction selected  <i>Note: SRI-Bus: Block Transfer Request - 4 Transfers (BTR4)</i>  <i>Note: SPB-Bus: not supported.</i></p> <p><math>110_B</math> Reserved.</p> <p><math>111_B</math> Reserved.</p>
<b>PATSEL</b>	26:24	rwh	<p><b>Pattern Select</b></p> <p>DMA channel TCS bit field to select the pattern detection operation (see Functional Description). If PATSEL[1:0] is not equal to <math>00_B</math> then a ME pattern detection operation defined by the channel data width (CHDW) will be performed. PATSEL[2] selects the pattern read register. If a pattern match is detected then a DMA channel interrupt shall be triggered.</p> <p><math>000_B</math> No pattern compare operation.</p> <p><math>001_B</math> DMA read move data compared with PRR0.      ...</p> <p><math>011_B</math> DMA read move data compared with PRR0.</p> <p><math>100_B</math> No pattern compare operation.</p> <p><math>101_B</math> DMA read move data compared with PRR1.      ...</p> <p><math>111_B</math> DMA read move data compared with PRR1.</p>
<b>SWAP</b>	27	rw	<p><b>Swap Data CRC Byte Order</b></p> <p>DMA channel TCS swap data CRC byte order.</p> <p><math>0_B</math> Byte order is not swapped</p> <p><math>1_B</math> Byte order is swapped.</p>

**Direct Memory Access (DMA)**

Field	Bits	Type	Description
<b>PRSEL</b>	28	rwh	<b>Peripheral Request Select</b> DMA channel TCS control bit field to select the source of a DMA request. $0_B$ DMA hardware request selected. $1_B$ DMA daisy chain request selected.
<b>0</b>	15:14, 31:29	r	<b>Reserved</b> Read as 0; must be written with 0.

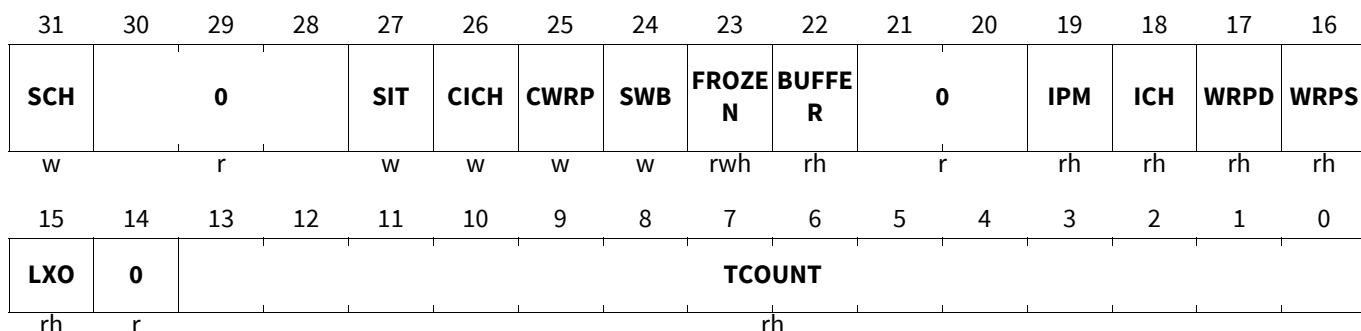
**DMARAM Channel c Shadow Address Register****SHADRc (c=000-127)****DMARAM Channel c Shadow Address Register( $2018_H + c * 20_H$ )****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>SHADR</b>	31:0	rwh	<b>Shadowed Address</b> 32-bit shadow address.

**DMARAM Channel c Control and Status Register**

The CHCSR bit fields are used for two functions:

1. Storing DMA channel status bits.
2. Write only triggers to set and clear DMA channel configuration and status.

**CHCSRc (c=000-127)****DMARAM Channel c Control and Status Register( $201C_H + c * 20_H$ )****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>TCOUNT</b>	13:0	rh	<b>Transfer Count</b> DMA channel status transfer count updated after DMARAM write back.

**Direct Memory Access (DMA)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>LXO</b>	15	rh	<p><b>Old Value of Pattern Detection</b></p> <p>DMA channel status bit to store the result of a pattern detection operation when 8-bit or 16-bit data width is selected.</p> <p><math>0_B</math> The corresponding pattern compare operation did not find a pattern match during the previous DMA read move.</p> <p><math>1_B</math> The corresponding pattern compare operation found a pattern match during the previous DMA read move.</p>
<b>WRPS</b>	16	rh	<p><b>Wrap Source Buffer</b></p> <p>Status bit indicates that a DMA channel has reached a wrap source buffer boundary. Bit is reset by software (DMA channel CHCSR.CWRP = <math>1_B</math> or DMA channel reset TSR.RST = <math>1_B</math>).</p> <p><math>0_B</math> No wrap source buffer occurred.</p> <p><math>1_B</math> Wrap source buffer occurred.</p>
<b>WRPD</b>	17	rh	<p><b>Wrap Destination Buffer</b></p> <p>Status bit indicates that a DMA channel has reached a wrap destination buffer boundary. Bit is reset by software (DMA channel CHCSR.CWRP = <math>1_B</math> or DMA channel reset TSR.RST = <math>1_B</math>).</p> <p><math>0_B</math> No wrap destination buffer occurred.</p> <p><math>1_B</math> Wrap destination buffer occurred.</p>
<b>ICH</b>	18	rh	<p><b>Interrupt from Channel</b></p> <p>As soon as ME CHSR.TCOUNT decrements, if ADICR.INTCT[0] = 0 and CHSR.COUNT = IRDV then ME CHSR.ICH is set else ME CHSR.ICH is set for each decrement of CSR.TCOUNT. Bit is reset by software (DMA channel CHCSR.CICH = <math>1_B</math> or by a DMA channel reset TSR.RST = <math>1_B</math>).</p> <p><math>0_B</math> An interrupt from channel has not been detected.</p> <p><math>1_B</math> An interrupt from channel has been detected.</p>
<b>IPM</b>	19	rh	<p><b>Pattern Detection from Channel</b></p> <p>Status bit indicates that a pattern match has been detected for the DMA channel when pattern detection is enabled. This bit is reset by software (DMA channel CHCSR.CICH = <math>1_B</math> or DMA channel reset TSR.RST = <math>1_B</math>).</p> <p><math>0_B</math> A pattern match has not been detected.</p> <p><math>1_B</math> A pattern match has been detected.</p>
<b>BUFFER</b>	22	rh	<p><b>DMA Double Buffering Active Buffer</b></p> <p>During a DMA double buffering operation, the status bit indicates which buffer is read or filled.</p> <p><math>0_B</math> Buffer 0 read or filled by DMA.</p> <p><math>1_B</math> Buffer 1 read or filled by DMA.</p>
<b>FROZEN</b>	23	rwh	<p><b>DMA Double Buffering Frozen Buffer</b></p> <p>If a DMA channel is configured for double buffering operation, the FROZEN bit indicates that one of the buffers is frozen and available for processing by a cyclic software task.</p> <p><i>Note:</i> <i>FROZEN bit shall only be set by the DMA and shall only be cleared by software.</i></p> <p><math>0_B</math> Buffer is not frozen.</p> <p><math>1_B</math> Buffer is frozen.</p>

**Direct Memory Access (DMA)**

Field	Bits	Type	Description
<b>SWB</b>	24	w	<p><b>DMA Double Buffering Switch Buffer</b>  When DMA double buffering is configured, the control bit is used to redirect data from one buffer to the other buffer.</p> <p><i>Note:</i> If a DMALL, ACCLL, SAFLL or CONLL operation is configured then SWB shall be 0<sub>B</sub>.</p> <p>0<sub>B</sub> No action.  1<sub>B</sub> Switch from buffer.</p>
<b>CWRP</b>	25	w	<p><b>Clear Wrap Buffer Interrupt</b>  Software clear of the DMA channel source and destination wrap buffer flags stored at CHCSR.WRPS and CHCSR.WRAPD. If the DMA channel is active in a ME then clear ME bit fields CHSR.WRPS and CHSR.WRAPD. Reading this bit returns a 0.</p> <p>0<sub>B</sub> No action.  1<sub>B</sub> Clear DMA channel bits CSR.WRPS and CSR.WRAPD.</p>
<b>CICH</b>	26	w	<p><b>Clear Interrupt for DMA Channel</b>  Software clear of the DMA channel flags stored at CHCSR.ICH and CHCSR.IPM. If the DMA channel is active in a ME then clear ME bit fields CHSR.ICH and CHSR.IPM. Reading this bit returns a 0.</p> <p>0<sub>B</sub> No action.  1<sub>B</sub> Clear DMA channel bits CSR.ICH and CSR.IPM.</p>
<b>SIT</b>	27	w	<p><b>Set Interrupt Trigger for DMA Channel</b>  Reading this bit returns a 0.</p> <p><i>Note:</i> If a DMALL, ACCLL, SAFLL or CONLL operation is configured then SIT must be 0<sub>B</sub>.</p> <p>0<sub>B</sub> No action.  1<sub>B</sub> DMA channel interrupt trigger will be activated.</p>
<b>SCH</b>	31	w	<p><b>Set Transaction Request</b>  Reading this bit returns a 0.</p> <p>0<sub>B</sub> No action.  1<sub>B</sub> DMA software request initiated for DMA channel. When SCH is set, DMA channel TSR.CH is set to indicate a DMA request is pending.</p>
<b>0</b>	14, 21:20, 30:28	r	<p><b>Reserved</b>  Read as 0; must be written with 0.</p>

## Direct Memory Access (DMA)

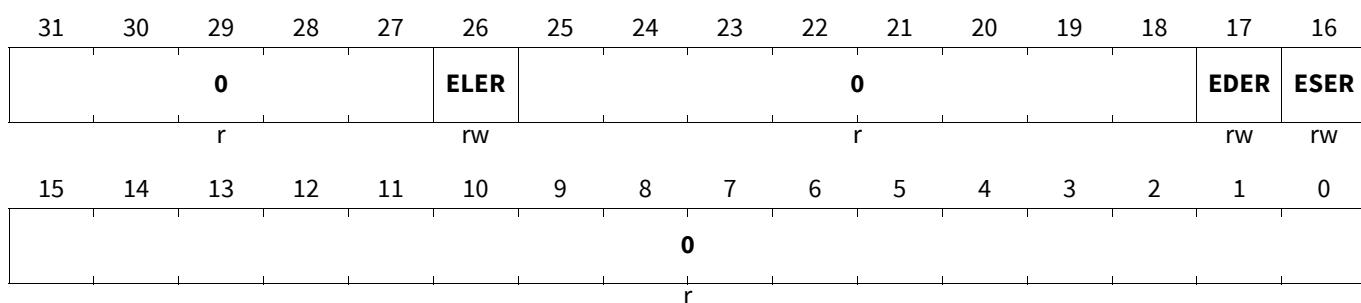
### 18.4.6 ME Registers

#### ME m Enable Error Register

The Enable Error Register enables an error interrupt service request for ME errors.

##### EER<sub>m</sub> (m=0-1)

**ME m Enable Error Register** **(0120<sub>H</sub>+m\*1000<sub>H</sub>)** **Application Reset Value: 0403 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>ESER</b>	16	rw	<b>Enable ME Source Error</b> This bit enables the generation of a ME source error interrupt. $0_B$ ME source error interrupt is disabled. $1_B$ ME source error interrupt is enabled.
<b>EDER</b>	17	rw	<b>Enable ME Destination Error</b> This bit enables the generation of a ME destination error interrupt. $0_B$ ME destination error interrupt is disabled. $1_B$ ME destination error interrupt is enabled.
<b>ELER</b>	26	rw	<b>Enable ME DMA Linked List Error</b> This bit enables the generation of a ME DMA Linked List error interrupt. $0_B$ ME DMA Linked List error interrupt is disabled. $1_B$ ME DMA Linked List error interrupt is enabled.
<b>0</b>	15:0, 25:18, 31:27	r	<b>Reserved</b> Read as 0; must be written with 0.

#### ME m Error Status Register

The Error Status Register indicates if the DMA controller could not service a DMA request. It indicates that an SPB Bus access or an SRI Bus access has terminated with errors.

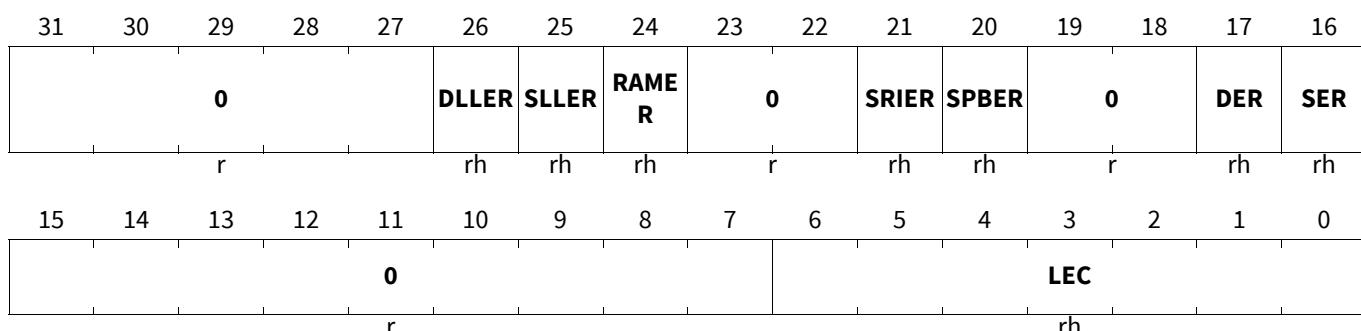
## Direct Memory Access (DMA)

### ERRSRm (m=0-1)

#### ME m Error Status Register

(0124<sub>H</sub>+m\*1000<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
LEC	6:0	rh	<b>ME Last Error Channel</b> This bit field indicates the DMA channel number of the last DMA channel of ME generating an error i.e. RAM error DMA_ERRSRm.RAMER, Safe Linked List error DMA_ERRSRm.SLLER, DMA Linked List error DMA_ERRSRm.DLLER and all on chip bus errors.
SER	16	rh	<b>ME Source Error</b> This bit is set whenever a ME error occurred during a source (read) move of a DMA transfer, or a request could not be serviced due to the access protection. 0 <sub>B</sub> No ME source error has occurred. 1 <sub>B</sub> ME source error has occurred.
DER	17	rh	<b>ME Destination Error</b> This bit is set whenever a ME error occurred during a destination (write) move of a DMA transfer, or a request could not be serviced due to the access protection. 0 <sub>B</sub> No ME destination error has occurred. 1 <sub>B</sub> ME destination error has occurred.
SPBER	20	rh	<b>ME SPB Bus Error</b> This bit is set when a ME DMA Move that has been started by the DMA SPB master interface leads to an error on the SPB Bus. 0 <sub>B</sub> No error occurred. 1 <sub>B</sub> An error occurred on SPB Bus interface.
SRIER	21	rh	<b>ME SRI Bus Error</b> This bit is set when a ME DMA Move that has been started by the DMA SRI master interface leads to an error on the SRI Bus. 0 <sub>B</sub> No error occurred. 1 <sub>B</sub> An error occurred on SRI Bus interface.
RAMER	24	rh	<b>ME RAM Error</b> This bit is set whenever a ME error occurred during the loading of a TCS from the DMARAM to the ME channel registers. 0 <sub>B</sub> No error occurred. 1 <sub>B</sub> An error occurred during the load of a TCS.

## Direct Memory Access (DMA)

Field	Bits	Type	Description
<b>SLLER</b>	25	rh	<b>ME Safe Linked List Error</b> This bit is set when a ME error occurred during the comparison of a SDCRC checksums during a safe linked list. 0 <sub>B</sub> No error occurred. 1 <sub>B</sub> An error occurred during the SDCRC checksum comparison.
<b>DLLER</b>	26	rh	<b>ME DMA Linked List Error</b> This bit is set when a ME error occurred during a DMALL, ACPLL, SAFLL or CONLL operation when a new TCS is loaded from anywhere in memory to overwrite the current TCS stored in the DMARAM. 0 <sub>B</sub> No error occurred. 1 <sub>B</sub> An error occurred during the loading of a new TCS.
<b>0</b>	15:7, 19:18, 23:22, 31:27	r	<b>Reserved</b> Read as 0; must be written with 0.

### ME m Clear Error Register

The Clear Error Register contains bits to clear the corresponding ME error flags.

#### CLREm (m=0-1)

#### ME m Clear Error Register

(0128<sub>H</sub>+m\*1000<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
					<b>CDLLE</b> R	<b>CSLLE</b> R	<b>CRAM</b> ER		<b>0</b>	<b>CSRIE</b> R	<b>CSPBE</b> R		<b>0</b>	<b>CDER</b>	<b>CSER</b>
				r	w	w	w	r	w	w	r	r	w	w	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								<b>0</b>				r			

Field	Bits	Type	Description
<b>CSER</b>	16	w	<b>Clear ME Source Error</b> 0 <sub>B</sub> No action 1 <sub>B</sub> Clear source error flag DMA_ERRSRm.SER.
<b>CDER</b>	17	w	<b>Clear ME Destination Error</b> 0 <sub>B</sub> No action 1 <sub>B</sub> Clear destination error flag DMA_ERRSRm.DER.
<b>CSPBER</b>	20	w	<b>Clear SPB Error</b> 0 <sub>B</sub> No action 1 <sub>B</sub> Clear error flag DMA_ERRSRm.SPBER.
<b>CSRIER</b>	21	w	<b>Clear SRI Error</b> 0 <sub>B</sub> No action 1 <sub>B</sub> Clear error flag DMA_ERRSRm.SRIER.

## Direct Memory Access (DMA)

Field	Bits	Type	Description
<b>CRAMER</b>	24	w	<b>Clear RAM Error</b> $0_B$ No action $1_B$ Clear error flag DMA_ERRSRm.RAMER.
<b>CSLLER</b>	25	w	<b>Clear SLL Error</b> $0_B$ No action $1_B$ Clear error flag DMA_ERRSRm.SLLER.
<b>CDLLER</b>	26	w	<b>Clear DLL Error</b> $0_B$ No action $1_B$ Clear error flag DMA_ERRSRm.DLLER.
<b>0</b>	15:0, 19:18, 23:22, 31:27	r	<b>Reserved</b> Read as 0; must be written with 0.

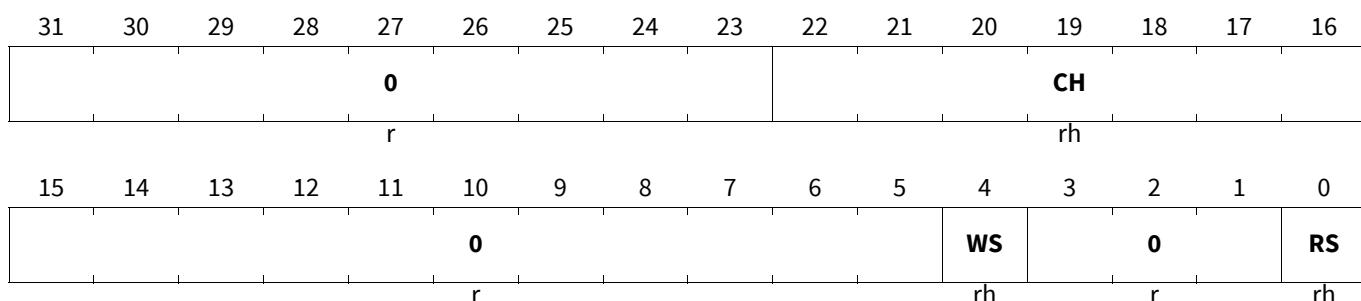
### ME m Status Register

The ME Status Register holds status information about the DMA transaction handled by the ME.

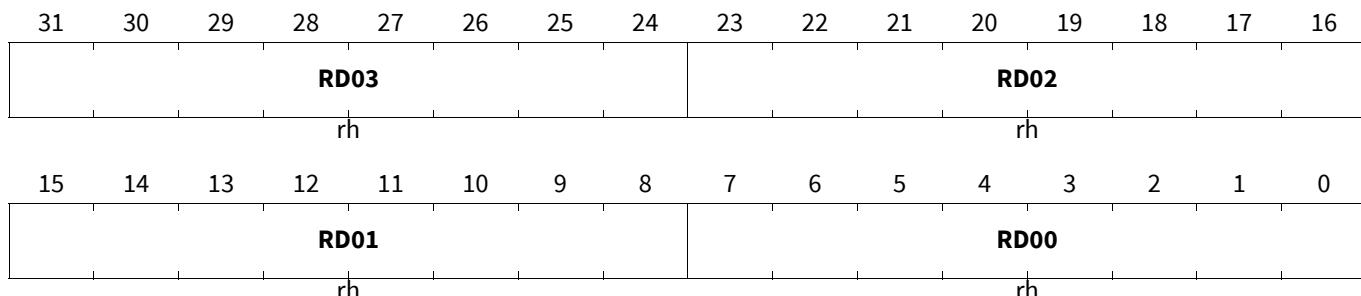
#### ME<sub>m</sub> Status Register

(0130<sub>H</sub>+m\*1000<sub>H</sub>)

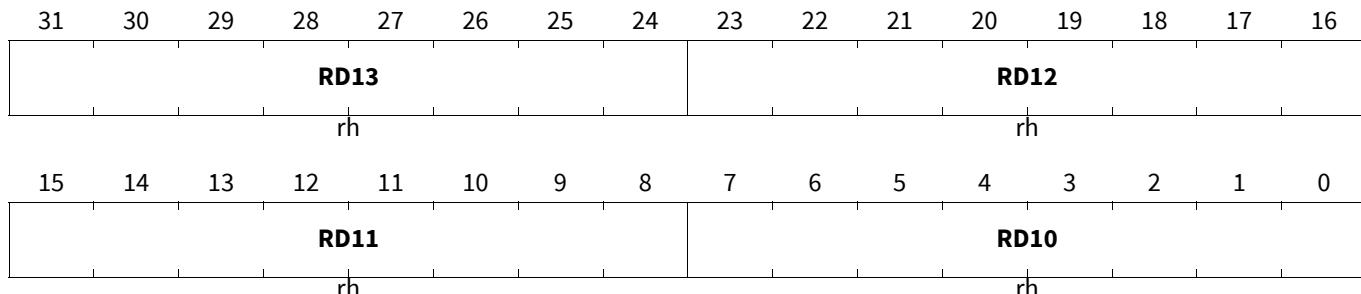
Application Reset Value: 0000 0000<sub>H</sub>



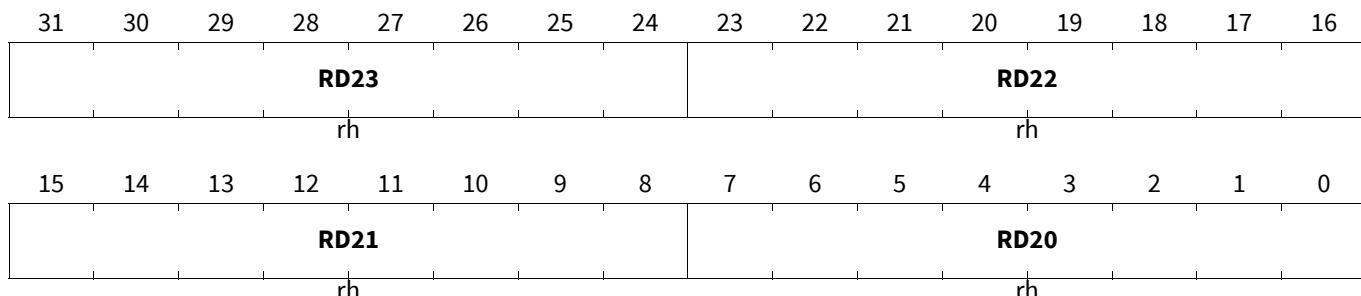
Field	Bits	Type	Description
<b>RS</b>	0	rh	<b>ME Read Status</b> $0_B$ ME is not performing a DMA read move. $1_B$ ME is performing a DMA read move.
<b>WS</b>	4	rh	<b>ME Write Status</b> $0_B$ ME is not performing a DMA write move. $1_B$ ME is performing a DMA write move.
<b>CH</b>	22:16	rh	<b>ME Active Channel</b> Indicates the number of the DMA Channel currently processed by the ME.
<b>0</b>	3:1, 15:5, 31:23	r	<b>Reserved</b> Read as 0; must be written with 0.

**Direct Memory Access (DMA)****ME m Read Register 0****MEm0R (m=0-1)****ME m Read Register 0****(0140<sub>H</sub>+m\*1000<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

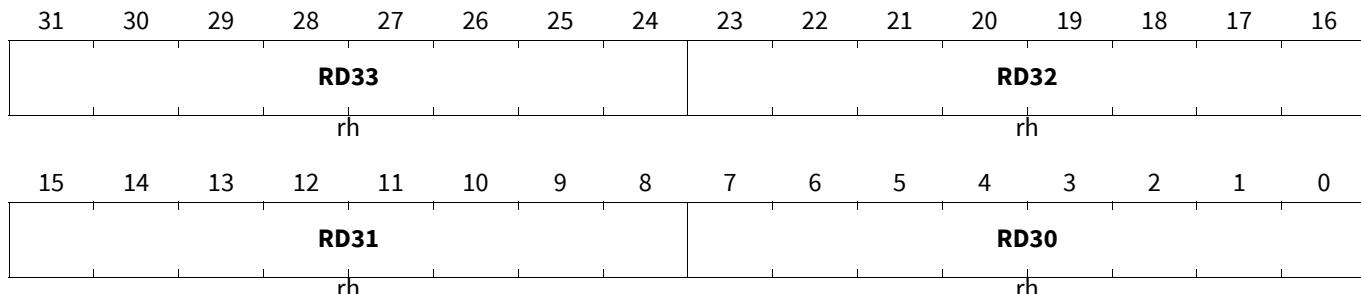
Field	Bits	Type	Description
<b>RD00</b>	7:0	rh	<b>DMA Read Move Data Byte</b>
<b>RD01</b>	15:8	rh	<b>DMA Read Move Data Byte</b>
<b>RD02</b>	23:16	rh	<b>DMA Read Move Data Byte</b>
<b>RD03</b>	31:24	rh	<b>DMA Read Move Data Byte</b>

**ME m Read Register 1****MEm1R (m=0-1)****ME m Read Register 1****(0144<sub>H</sub>+m\*1000<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

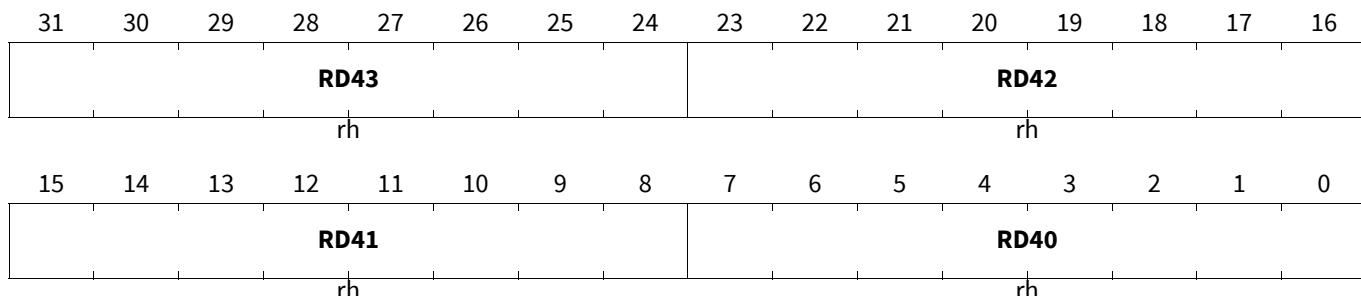
Field	Bits	Type	Description
<b>RD10</b>	7:0	rh	<b>DMA Read Move Data Byte</b>
<b>RD11</b>	15:8	rh	<b>DMA Read Move Data Byte</b>
<b>RD12</b>	23:16	rh	<b>DMA Read Move Data Byte</b>
<b>RD13</b>	31:24	rh	<b>DMA Read Move Data Byte</b>

**Direct Memory Access (DMA)****ME m Read Register 2****MEm2R (m=0-1)****ME m Read Register 2****(0148<sub>H</sub>+m\*1000<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

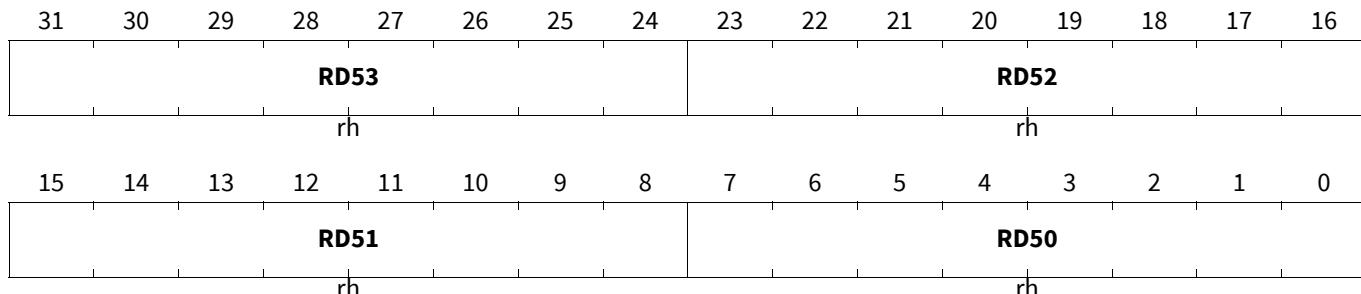
Field	Bits	Type	Description
<b>RD20</b>	7:0	rh	<b>DMA Read Move Data Byte</b>
<b>RD21</b>	15:8	rh	<b>DMA Read Move Data Byte</b>
<b>RD22</b>	23:16	rh	<b>DMA Read Move Data Byte</b>
<b>RD23</b>	31:24	rh	<b>DMA Read Move Data Byte</b>

**ME m Read Register 3****MEm3R (m=0-1)****ME m Read Register 3****(014C<sub>H</sub>+m\*1000<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>RD30</b>	7:0	rh	<b>DMA Read Move Data Byte</b>
<b>RD31</b>	15:8	rh	<b>DMA Read Move Data Byte</b>
<b>RD32</b>	23:16	rh	<b>DMA Read Move Data Byte</b>
<b>RD33</b>	31:24	rh	<b>DMA Read Move Data Byte</b>

**Direct Memory Access (DMA)****ME m Read Register 4****MEM4R (m=0-1)****ME m Read Register 4****(0150<sub>H</sub>+m\*1000<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>RD40</b>	7:0	rh	<b>DMA Read Move Data Byte</b>
<b>RD41</b>	15:8	rh	<b>DMA Read Move Data Byte</b>
<b>RD42</b>	23:16	rh	<b>DMA Read Move Data Byte</b>
<b>RD43</b>	31:24	rh	<b>DMA Read Move Data Byte</b>

**ME m Read Register 5****MEM5R (m=0-1)****ME m Read Register 5****(0154<sub>H</sub>+m\*1000<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>RD50</b>	7:0	rh	<b>DMA Read Move Data Byte</b>
<b>RD51</b>	15:8	rh	<b>DMA Read Move Data Byte</b>
<b>RD52</b>	23:16	rh	<b>DMA Read Move Data Byte</b>
<b>RD53</b>	31:24	rh	<b>DMA Read Move Data Byte</b>

## Direct Memory Access (DMA)

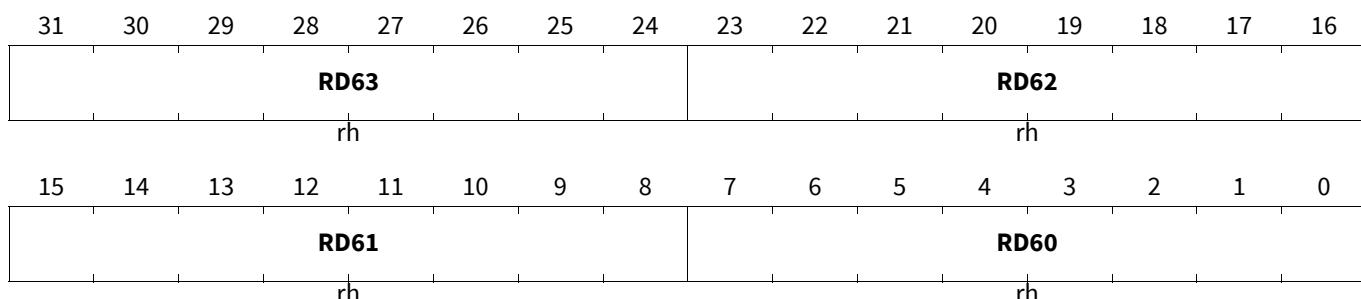
### ME m Read Register 6

**MEm6R (m=0-1)**

#### ME m Read Register 6

( $0158_H + m * 1000_H$ )

Application Reset Value:  $0000\ 0000_H$



Field	Bits	Type	Description
<b>RD60</b>	7:0	rh	<b>DMA Read Move Data Byte</b>
<b>RD61</b>	15:8	rh	<b>DMA Read Move Data Byte</b>
<b>RD62</b>	23:16	rh	<b>DMA Read Move Data Byte</b>
<b>RD63</b>	31:24	rh	<b>DMA Read Move Data Byte</b>

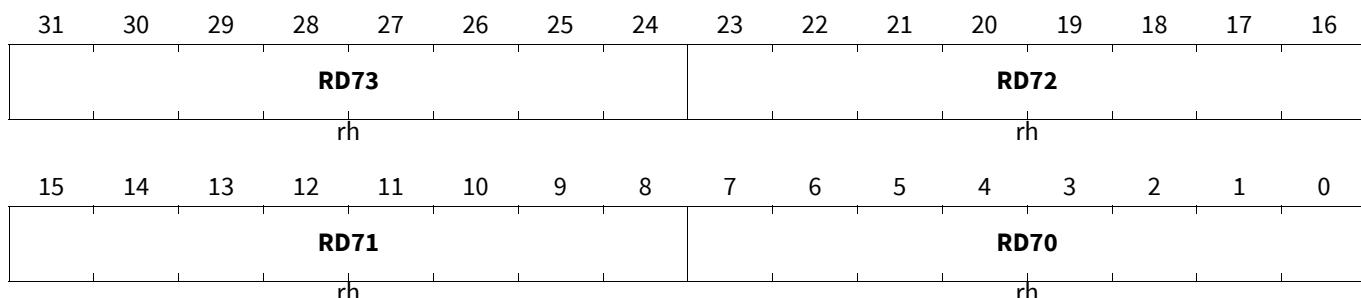
### ME m Read Register 7

**MEm7R (m=0-1)**

#### ME m Read Register 7

( $015C_H + m * 1000_H$ )

Application Reset Value:  $0000\ 0000_H$



Field	Bits	Type	Description
<b>RD70</b>	7:0	rh	<b>DMA Read Move Data Byte</b>
<b>RD71</b>	15:8	rh	<b>DMA Read Move Data Byte</b>
<b>RD72</b>	23:16	rh	<b>DMA Read Move Data Byte</b>
<b>RD73</b>	31:24	rh	<b>DMA Read Move Data Byte</b>

### ME m Channel Read Data CRC Register

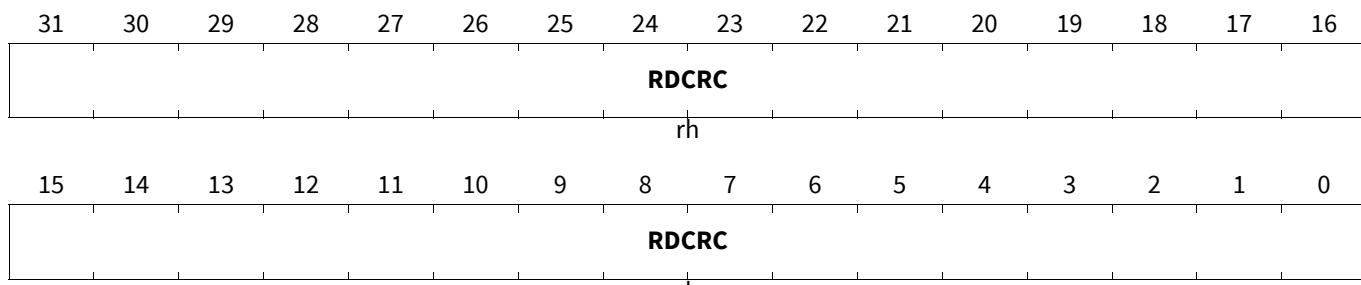
The read only DMA active channel Read Data CRC Register will store one polynomial calculation during each DMA read move provided that there is no retry or error condition flagged. In order to start a CRC32 sequence the MEmRDCRC must be initialized (e.g. written with  $00000000_H$  or with a desired start value) and an DMA transaction must be set up (start address, length, etc.).

## Direct Memory Access (DMA)

### MEmRDCRC (m=0-1)

**ME m Channel Read Data CRC Register (0180<sub>H</sub>+m\*1000<sub>H</sub>)**

**Application Reset Value: 0000 0000<sub>H</sub>**



### MEmSDCRC (m=0-1)

**ME m Channel Source and Destination Address CRC Register(0184<sub>H</sub>+m\*1000<sub>H</sub>)**

**Application Reset Value:**

**0000 0000<sub>H</sub>**

### ME m Channel Source and Destination Address CRC Register

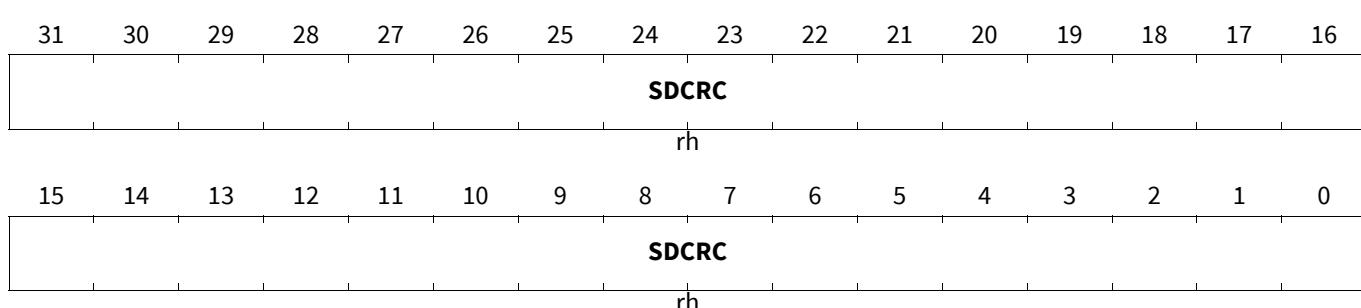
The read only DMA active channel Source and Destination Address CRC Register will store one polynomial calculation during each DMA read and each DMA write move provided there is no retry or error condition flagged.

### MEmSDCRC (m=0-1)

**ME m Channel Source and Destination Address CRC Register(0184<sub>H</sub>+m\*1000<sub>H</sub>)**

**Application Reset Value:**

**0000 0000<sub>H</sub>**



### ME m Channel Source Address Register

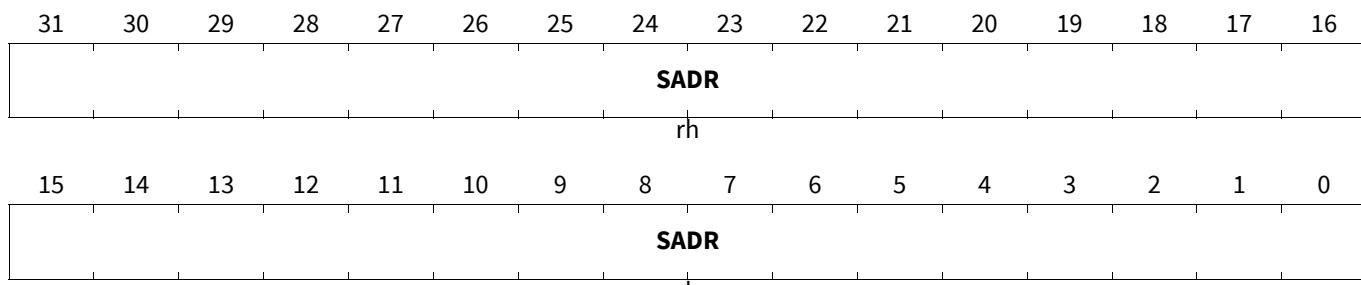
The read only DMA active channel Source Address Register holds the 32-bit source address. If a DMA channel is active, MEmSADR is updated continuously and shows the actual source address used for DMA read moves.

## Direct Memory Access (DMA)

### MEmSADR (m=0-1)

**ME m Channel Source Address Register (0188<sub>H</sub>+m\*1000<sub>H</sub>)**

**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
SADR	31:0	rh	<b>Source Address</b> Active DMA channel 32-bit source address used for DMA read moves.

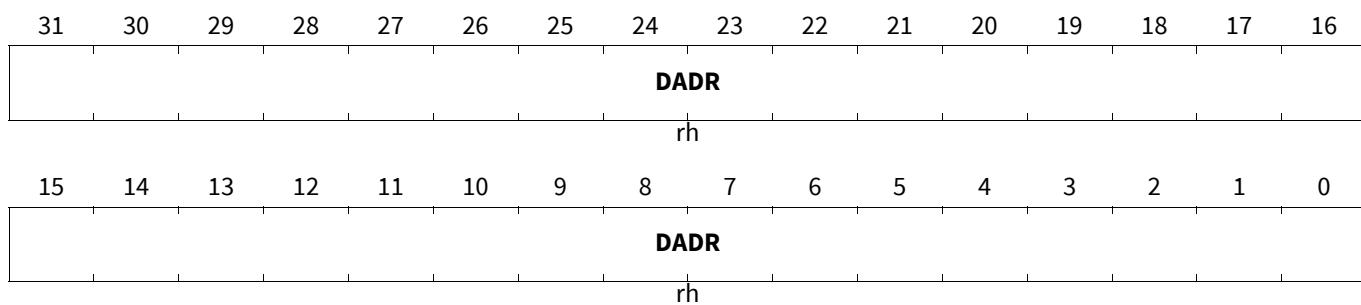
### ME m Channel Destination Address Register

The read only DMA active channel Destination Address Register holds the 32-bit destination address. If a DMA channel is active, MEmDADR is updated continuously and shows the actual destination address used for DMA write moves.

### MEmDADR (m=0-1)

**ME m Channel Destination Address Register(018C<sub>H</sub>+m\*1000<sub>H</sub>)**

**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
DADR	31:0	rh	<b>Destination Address</b> Active DMA channel 32-bit destination address used for DMA write moves.

### ME m Channel Address and Interrupt Control Register

The read only DMA active channel Address and Interrupt Control Register controls how source and destination addresses are updated after a DMA move. It determines whether or not a source or destination address register update is shadowed. The interrupt control bits enable the generation of DMA channel interrupt triggers.

## Direct Memory Access (DMA)

### MEmADICR (m=0-1)

**ME m Channel Address and Interrupt Control Register(0190<sub>H</sub>+m\*1000<sub>H</sub>)**

0000<sub>H</sub>

**Application Reset Value: 0000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>IRDV</b>			<b>INTCT</b>			<b>WRPD E</b>	<b>WRPS E</b>	<b>0</b>	<b>STAM P</b>	<b>DCBE</b>	<b>SCBE</b>	<b>SHCT</b>			
rh			rh	rh	rh	rh	rh	r	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>CBLD</b>			<b>CBLS</b>			<b>INCD</b>	<b>DMF</b>			<b>INCS</b>	<b>SMF</b>				
rh			rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
<b>SMF</b>	2:0	rh	<b>Source Address Modification Factor</b> Active DMA channel source address modification factor.
<b>INCS</b>	3	rh	<b>Increment of Source Address</b> Active DMA channel increment of source address control.
<b>DMF</b>	6:4	rh	<b>Destination Address Modification Factor</b> Active DMA channel destination address modification factor.
<b>INCD</b>	7	rh	<b>Increment of Destination Address</b> Active DMA channel increment of destination address control.
<b>CBLS</b>	11:8	rh	<b>Circular Buffer Length Source</b> Active DMA channel circular source buffer control.
<b>CBLD</b>	15:12	rh	<b>Circular Buffer Length Destination</b> Active DMA channel circular destination buffer control.
<b>SHCT</b>	19:16	rh	<b>Shadow Control</b> Active DMA channel control of shadow address register function.
<b>SCBE</b>	20	rh	<b>Source Circular Buffer Enable</b> Active DMA channel circular source buffer enable/disable.
<b>DCBE</b>	21	rh	<b>Destination Circular Buffer Enable</b> Active DMA channel circular destination buffer enable/disable.
<b>STAMP</b>	22	rh	<b>Time Stamp</b> Active DMA channel control to enable the appendage of a timestamp after the end of the last DMA Move during a DMA transaction.
<b>WRPSE</b>	24	rh	<b>Wrap Source Enable</b> Active DMA channel source buffer interrupt trigger enable/disable.
<b>WRPDE</b>	25	rh	<b>Wrap Destination Enable</b> Active DMA channel destination buffer interrupt trigger enable/disable.
<b>INTCT</b>	27:26	rh	<b>Interrupt Control</b> Active DMA channel interrupt service request control.
<b>IRDV</b>	31:28	rh	<b>Interrupt Raise Detect Value</b> Active DMA channel control of the Threshold Limit of DMA channel CHSR.TCOUNT for triggering a channel interrupt service request.

## Direct Memory Access (DMA)

Field	Bits	Type	Description
0	23	r	<b>Reserved</b> Read as 0; must be written with 0.

### ME m Channel Control Register

The read only DMA active channel Channel Control Register contains the configuration and control bits.

#### MEMCHCR (m=0-1)

ME m Channel Control Register														(0194 <sub>H</sub> +m*1000 <sub>H</sub> )				Application Reset Value: 0000 0000 <sub>H</sub>														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																0																

Field	Bits	Type	Description
TREL	13:0	rh	<b>Transfer Reload Value</b> Active DMA channel Transfer Reload Value.
BLKM	18:16	rh	<b>Block Mode</b> Active DMA channel Block Mode.
RROAT	19	rh	<b>Reset Request Only After Transaction</b> Active DMA channel Reset Request Only After Transaction.
CHMODE	20	rh	<b>Channel Operation Mode</b> Active DMA channel Channel Operation Mode.
CHDW	23:21	rh	<b>Channel Data Width</b> Active DMA channel DMA move Channel Data Width.
PATSEL	26:24	rh	<b>Pattern Select</b> Active DMA channel Pattern Select control.
SWAP	27	rh	<b>Swap Data CRC byte order</b> Active DMA channel swap data CRC byte order.
PRSEL	28	rh	<b>Peripheral Request Select</b> Active DMA channel Peripheral Request Select.
0	15:14, 31:29	r	<b>Reserved</b> Read as 0; must be written with 0.

### ME m Channel Shadow Address Register

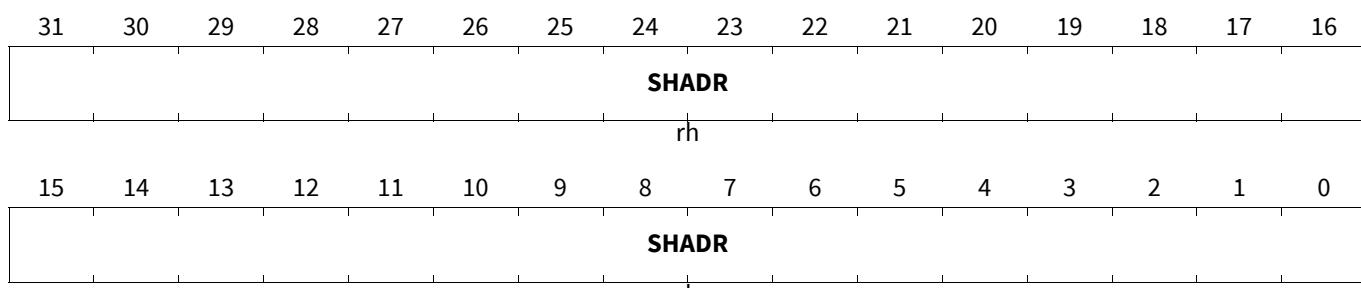
The read only DMA active channel Shadow Address Register holds the 32-bit shadow address.

## Direct Memory Access (DMA)

### MEMSHADR (m=0-1)

**ME m Channel Shadow Address Register (0198<sub>H</sub>+m\*1000<sub>H</sub>)**

**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>SHADR</b>	31:0	rh	<b>Shadowed Address</b> This bit field holds the 32-bit shadow address of the active DMA channel. The function of the shadow address is set by the shadow control settings.

### ME m Channel Status Register

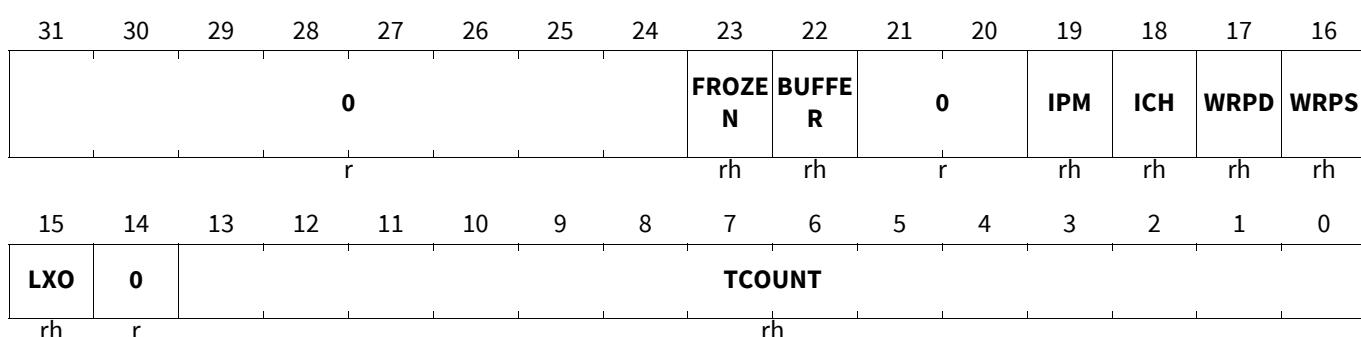
The read only DMA active channel Channel Status Register contains the current transfer count, pattern detection compare result and the status of traffic management interrupt triggers.

### MEMCHSR (m=0-1)

**ME m Channel Status Register**

(019C<sub>H</sub>+m\*1000<sub>H</sub>)

**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>TCOUNT</b>	13:0	rh	<b>Transfer Count Status</b> Active DMA channel count of the number of DMA transfers. TCOUNT is loaded with the DMA channel value of CHCFGR.TREL when TSR.CH becomes set (and TCOUNT = 0). After each DMA transfer, TCOUNT is decremented by 1.
<b>LXO</b>	15	rh	<b>Old Value of Pattern Detection</b> Active DMA channel compare result of a pattern compare operation when 8-bit or 16-bit data width is selected.
<b>WRPS</b>	16	rh	<b>Wrap Source Buffer</b> Active DMA channel Wrap Source Buffer status bit.
<b>WRPD</b>	17	rh	<b>Wrap Destination Buffer</b> Active DMA channel Wrap Destination Buffer status bit.

**Direct Memory Access (DMA)**

Field	Bits	Type	Description
<b>ICH</b>	18	rh	<b>Interrupt from Channel</b> Active DMA channel detection of channel interrupt service request.
<b>IPM</b>	19	rh	<b>Pattern Detection from Channel</b> Active DMA channel detection of pattern match interrupt service request.
<b>BUFFER</b>	22	rh	<b>DMA Double Buffering Active Buffer</b> Active DMA channel DMA Double Buffering Active Buffer status bit. $0_B$ Buffer 0 read or filled by DMA. $1_B$ Buffer 1 read or filled by DMA.
<b>FROZEN</b>	23	rh	<b>DMA Double Buffering Frozen Buffer</b> Active DMA channel DMA Double Buffering Frozen Buffer status bit.
<b>0</b>	14, 21:20, 31:24	r	<b>Reserved</b> Read as 0; must be written with 0.

## Direct Memory Access (DMA)

### 18.5 Debug

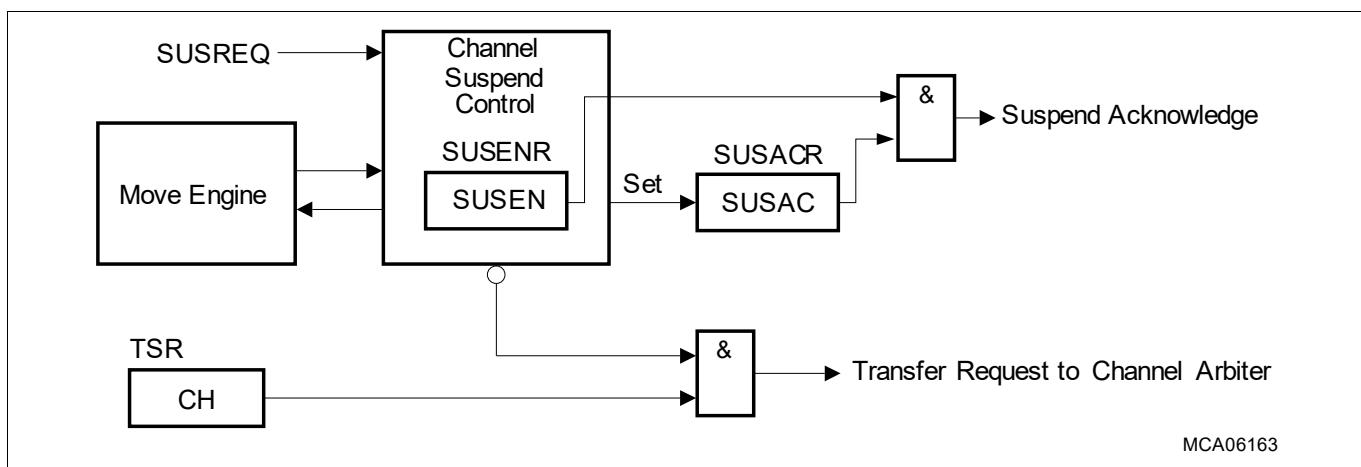
The DMA implements the following debug features:

#### 18.5.1 DMA Channel Suspend

Two DMA channel registers (at [Figure 224](#)) control and report the channel suspend operation:

- The Suspend Enable Register (SUSENR) enables and disables DMA channel soft suspend mode capability.
- The Suspend Acknowledge Register (SUSACR) indicates the DMA channel soft suspend status.

The On Chip Debug System (OCDS) is able to generate a DMA suspend request (SUSREQ). When the suspend request becomes active, the state of a DMA channel becomes frozen to ensure that the state of the DMA channels can be analyzed by reading the DMA channel register contents. If a DMA channel is active then the current DMA transfer is completed. The DMA signals the suspend mode status back to the OCDS via a suspend acknowledge.



**Figure 224 DMA Channel Suspend**

#### Entering Channel Suspend Mode

A DMA channel is configured for Suspend Mode when the DMA channel suspend enable bit in the Suspend Mode Enable Register SUSENR.SUSEN is set. When SUSREQ becomes active, for DMA channels enabled for [DMA Channel Suspend](#), the operation is as follows:

- **Idle State, Reset State, Halt State and Pending State:** the DMA channel suspend active status flag SUSACR.SUSAC is set and the DMA channel state is frozen.
- **Active State:** the DMA channel suspend active status flag SUSACR.SUSAC is set on completion of the current DMA transfer and the DMA channel state is frozen.

The DMA suspend acknowledge flag becomes active when all DMA channels that are enabled for [DMA Channel Suspend](#) have set their DMA channel suspend active status flag SUSACR.SUSAC. In [DMA Channel Suspend](#), the DMA channel contents may be modified. These modifications are taken into account for further DMA transfers or DMA transactions of the related DMA channel after [DMA Channel Suspend](#) has been left.

DMA channels that are disabled for [DMA Channel Suspend](#) (SUSENR.SUSEN = 0) continue to execute the DMA channel operation.

#### Exiting Channel Suspend Mode

[DMA Channel Suspend](#) mode is left and the DMA channel operation is resumed if either the SUSREQ signal becomes inactive, or if the enable bit SUSENR.SUSEN is reset by software.

## Direct Memory Access (DMA)

### 18.5.2 Software Activation of DMA Channel Interrupt Service Requests

Each DMA channel interrupt service request shall be activated by programming the DMA channel CHCSR.SIT = 1<sub>B</sub>.

### 18.5.3 Software Activation of DMA RP Error Interrupt Service Requests

Each DMA RP error interrupt service request shall be activated by programming the RP ERRINT.SIT = 1<sub>B</sub>.

### 18.5.4 OCDS Trigger Bus (OTGB) Interface

A **TS** is a collection of signals which supports a specific debug use case. The OCDS Trigger Set Select (OTSS) register controls which TS is applied to one of two DMA OTGB interfaces: OTGB0 or OTGB1. A TS is routed by the OCDS Trigger Multiplexer (OTGM) to the OCDS Trigger Switch (OTGS).

#### DMA Trigger Sets

The DMA Trigger Sets (at [Table 586](#)) are selected and routed to the 16 bit OTGB0 or OTGB1 interface by programming the OTSS register. Only one TS shall be output at a time either on OTGB0 or on OTGB1. The OTGB0 and OTGB1 have no dependency on the OCDS enabled state.

DMA TS are unique to each DMA configuration and dependent on the number of DMA channels and ME.

**Table 586 DMA Trigger Sets**

Index	Description	Details
0	No Trigger Set selected	
1	<b>Channels (TS16_PF)</b> Active channels	<a href="#">Table 587</a>
2	<b>Channels (TS16_ERR)</b> Error channel number and flags	<a href="#">Table 588</a>
8	<b>Transaction Request State (TS16_C15)</b> Transaction Request State Channel [15:0]	
9	<b>Transaction Request State (TS16_C31)</b> Transaction Request State Channel [31:16]	
10	<b>Transaction Request State (TS16_C47)</b> Transaction Request State Channel [47:32]	
11	<b>Transaction Request State (TS16_C63)</b> Transaction Request State Channel [63:48]	
12	<b>Transaction Request State (TS16_C79)</b> Transaction Request State Channel [79:64]	
13	<b>Transaction Request State (TS16_C95)</b> Transaction Request State Channel [95:80]	
14	<b>Transaction Request State (TS16_C111)</b> Transaction Request State Channel [111:96]	
15	<b>Transaction Request State (TS16_C127)</b> Transaction Request State Channel [127:112]	
Other	Reserved. No Trigger Set selected	

## Direct Memory Access (DMA)

### Performance Trigger Set

The performance **TS** (TS16\_PF) continuously monitors ME activity. While a ME is servicing a **DMA Request**, the ME idle/active bit TS16\_PF.ME0/1 is set and the TS16\_PF.CH0/1 field is set to the number of the DMA channel. When the ME is idle the idle/active bit TS16\_PF.ME0/1 goes low. The TS16\_PF.CH0/1 bit field retains the value of the last active DMA channel.

**Table 587 TS16\_PF Trigger Set Channels**

Bits	Name	Description
[6:0]	CH0	Number of DMA channel active in ME0
7	ME0	ME0 idle/active 0 <sub>B</sub> ME0 is idle 1 <sub>B</sub> ME0 is active
[14:8]	CH1	Number of DMA channel active in ME1
15	ME1	ME1 active 0 <sub>B</sub> ME1 is idle 1 <sub>B</sub> ME1 is active

### Error Trigger Set

The error **TS** (TS16\_ERR) triggers an event for DMA source errors and DMA destination errors.

If a **ME** detects a **SER** or **DER**, the DMA sets TS16\_ERR as follows:

- If **MEm** detects a **SER**, the DMA sets TS16\_ERR.MEmSE for one trigger cycle.
- If **MEm** detects a **DER**, the DMA sets TS16\_ERR.MEmDE for one trigger cycle.
- The DMA sets TS16\_ERR.LEC to the last error channel for one trigger cycle. If both MEs generate a **SER** and/or **DER** in the same clock cycle, the DMA shall set TS16\_ERR.LEC to the ME0 last error channel.

**Table 588 TS16\_ERR Trigger Set Channels**

Bits	Name	Description
[6:0]	LEC	Last error channel number
[11:7]		Reserved
12	ME0SE	ME0 Source Error
13	ME0DE	ME0 Destination Error
14	ME1SE	ME1 Source Error
15	ME1DE	ME1 Destination Error

### Request Trigger Sets

The request trigger sets (TS16\_C15, etc.) continuously monitor the state of the DMA channel TSR.CH bits.

### 18.5.5 MCDS Trace Interface

Each DMA on chip bus master interface generates a trace vector as follows:

- One 8 bit vector from SPB master interface via BCU\_SPB to BAL\_SPB inside MCDS (i.e. spb\_clk domain).
- One 8 bit vector from SRI master interface via SRI\_XBAR to BAL\_SRI inside MCDS (i.e. sri\_clk domain).

The trace vector (at **Table 589**) identifies the DMA channel and ME making an on chip bus access.

## Direct Memory Access (DMA)

**Table 589 Trace Vector Definition**

Bits	Trace
[6:0]	DMA Channel
[7]	Move Engine 0 <sub>B</sub> ME0 1 <sub>B</sub> ME1

The trace vector is used for OCDS Level 3 and OCDS Level 1 purposes:

### DMA Trace Signal Generation for OCDS Level 3

The trace mechanism enables the MCDS system to trace transactions on the on chip bus generated by one or a group of dedicated DMA internal sources.

### DMA Trace Signal Generation for OCDS Level 1

For OCDS Level 1 purposes the DMA provides 8 bit trace vectors.

## Direct Memory Access (DMA)

### 18.6 Use Cases

#### 18.6.1 Move Operation

A code example is provided to give an overview of core DMA functionality. The example realizes DMA channel 000 executing a one word (32 bit-length) single data transfer from a source module to a data module without the intervention of the CPU. The DMA transaction is triggered by a DMA software request in this example. The DMA also supports DMA hardware requests. No interrupt or further functions are created in this example, please see the respective registers for more details.

##### 18.6.1.1 Step Description to Initialize and Trigger a DMA Transaction

- (Line 1) The 32-bit source address (DMA\_SADR000\_ADD) gets stored in the source address register DMA\_SADR000.
- (Line 2) The 32-bit destination address (DMA\_DADR000\_ADD) gets stored in the destination address register DMA\_DADR000.
- (Line 3) The channel data width is defined in the channel configuration register DMA\_CHCFGR000. Setting DMA\_CHCFGR000.[23:21] = 010<sub>B</sub> configures the channel data width to 32 bit.
- (Line 4) DMA hardware requests are disabled in the transaction state register DMA\_TSR000. Writing DMA\_TSR000.[17] = 1 disables the hardware transfer requests.
- (Line 5) This line starts the data transfer between the source and destination memory address.

*Note:* The declaration of DMA\_SADR000\_ADD and DMA\_DADR000\_ADD must be done by the user.

#### DMA channel 000 Code

```
(1) DMA_SADR000 = DMA_SADR000_ADD; // source address
(2) DMA_DADR000 = DMA_DADR000_ADD; // destination address
(3) DMA_CHCFGR000.U = (0x2 << 21); // channel data width
(4) DMA_TSR000.U |= (1 << 17); // disable DMA hardware requests
(5) DMA_CHCSR000.U |= (1 << 31); // initiate DMA software request
```

*Note:* The DMA transaction can be started anywhere in the program.

#### 18.6.2 Error Handler

Each RP generates a single error interrupt trigger to signal a number of different parallel error conditions:

- ME detected errors during the execution of DMA moves.
- TRL errors when a DMA request is not serviced.

The DMA RP error interrupt service request shall be serviced by the CPU supervising the RP.

An EH shall read the following registers to identify the error type and the DMA error channel:

- ME Error Status Registers (ERRSR) to detect the following:
  - Last Error Channel (LEC)
  - Source Error (SER)
  - Destination Error (DER)
  - RAM Error (RAMER)
  - Safe Linked List Error (SLLER)
  - DMA Linked List (DLLER)
- DMA channel Transaction State Registers (TSR) to detect the following:

## Direct Memory Access (DMA)

- Transaction Request State (CH)
- Transaction Request Lost (TRL)

The error routine should disable further DMA requests from the DMA error channel:

- DMA hardware requests: the CPU shall write DMA error channel TSR.DCH = 1<sub>B</sub>
- DMA software requests: the CPU shall not write DMA error channel CHCSR.SCH = 1<sub>B</sub>

The error routine should clear the error status flag(s) as follows:

- ME error flags:
  - SER: the CPU error routine shall write ME CLRE.CSER = 1<sub>B</sub>
  - DER: the CPU error routine shall write ME CLRE.CDER = 1<sub>B</sub>
  - SPBER: the CPU error routine shall write ME CLRE.CSPBER = 1<sub>B</sub>
  - SRIER: the CPU error routine shall write ME CLRE.CSRIER = 1<sub>B</sub>
  - RAMER: the CPU error routine shall write ME CLRE.CRAMER = 1<sub>B</sub>
  - SLLER: the CPU error routine shall write ME CLRE.CSLLER = 1<sub>B</sub>
  - DLLER: the CPU error routine shall write ME CLRE.CDLLER = 1<sub>B</sub>
- DMA channel error flags:
  - TRL: the CPU error routine shall write the DMA error channel TSR.CTL = 1<sub>B</sub>

The user shall identify the origin of the error. The DMA configuration shall be changed to prevent a reoccurrence of the error. On completion DMA requests shall be serviced as follows:

- DMA hardware requests: the CPU shall write DMA channel TSR.ECH = 1<sub>B</sub>
- DMA software requests: the CPU shall write DMA channel CHCSR.SCH = 1<sub>B</sub>

### 18.6.3 Data Communication

From TC39xB DMA and TC38xA DMA onwards, the DMA has been enhanced to report **TRL** errors if the DMA channel is disabled for hardware requests and a **DMA Hardware Request** is detected.

For example, if the application is transmitting data via a QSPI, the software may perform one write to the TXFIFO or may set the interrupt flag in the TXFIFO interrupt node to initiate data transmission. After each data word is loaded into the TXFIFO, the QSPI triggers a TXFIFO interrupt service request. The system effect is that (n + 1) interrupts are generated to move (n) data words. If the application uses the DMA to move data into the TXFIFO and the DMA channel is configured for **Single Mode**, the DMA will disable **DMA Hardware Request** after the DMA has serviced (n) interrupts and report a **TRL** for the last interrupt.

To prevent this **TRL** event, the software should configure the DMA channel as a **DMA Linked List (DMALL)** composed of the following DMA transactions:

- DMA Transaction 1: perform (n) DMA moves to write data to TXFIFO.
- DMA Transaction 2: perform a dummy DMA move to read and write data to an unused address in DSPIR.

If the DMA is used for data communication, the application should consider the possibility of **TRL** generation after the last data word.

### 18.7 Revision History

**Table 590 Changes**

Reference	Change to Previous Version	Comment
V0.1.15		
Page 59	Safety Flip-Flops <b>Chapter 18.4.1</b> added	

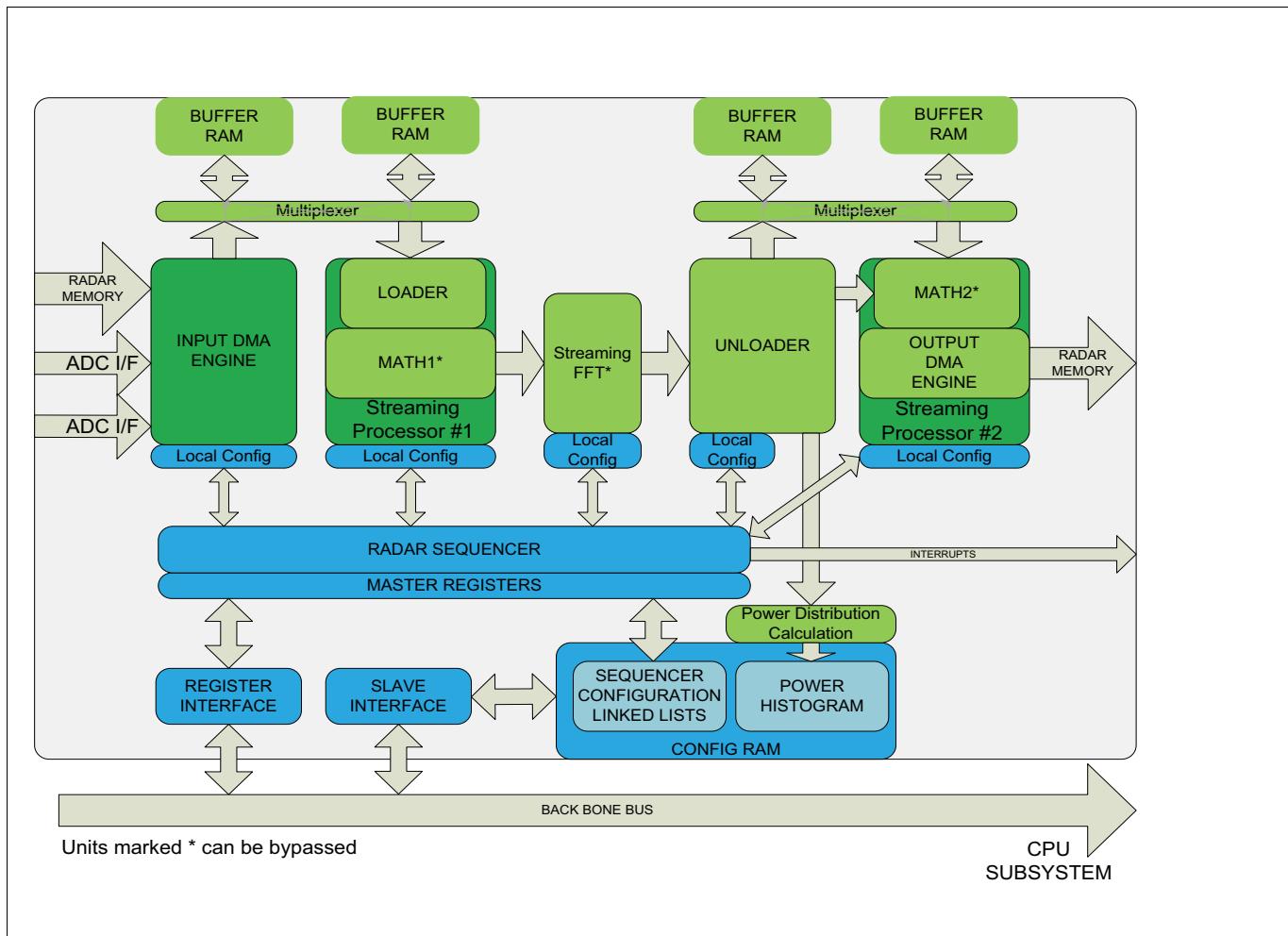
**Direct Memory Access (DMA)****Table 590 Changes**

Reference	Change to Previous Version	Comment
<a href="#">Page 7</a>	Figure “Software Control” updated	
<b>V0.1.16</b>		
<a href="#">Page 13</a>	Update Resetting a DMA Channel.	
<b>V0.1.17</b>		
<a href="#">Page 39</a>	Updated section “Active Buffer Full or Empty”.	
<a href="#">Page 55</a>	Updated section “Data Integrity Testing”.	

## Signal Processing Unit (SPU)

### 19 Signal Processing Unit (SPU)

The Signal Processing Unit (SPU) is a semi-autonomous accelerator for performing Fast Fourier Transforms (FFTs) on data from one or more dedicated ADC interfaces. The SPU uses a three stage, streaming architecture to provide data pre-processing, FFT, and data post-processing operations. The SPU uses the Radar Memory to store datasets and has internal buffer memories which are used to store the data currently progressing through the processing pipeline.



**Figure 225 SPU Architecture**

#### 19.1 Feature List

The radar SPU has the following top level features

- Up to 2 concurrent SPU instances active concurrently
- A maximum achievable performance can approach 1 clock per sample processed for large data cubes. (See [Section 19.7.14](#) of this document for a more detailed explanation of factors affecting SPU throughput.)
- A 256bit bus connection to Radar memory with 2 clocks per read or write access
- Radar sequencer to manage the automatic execution of linked list where execution units are reconfigured between computation steps
- Internal configuration memory for linked lists
- 2 selectable input data flows (ADC interface or main RAM)
  - The ADC Interface supports real or complex data

## Signal Processing Unit (SPU)

- Input / output precision selectable: 16 or 32bits
- Input DMA with HW transpose unit and data padding
- Multiple execution units
  - Windowing with complex windowing and HW support for phase demodulation
  - FFT from 8 to 2048 points
  - 2 x 1D-CFAR, thresholding units implementing 4 CA and 3 GOS algorithms
  - 1 x BIN rejection unit
  - Non coherent integration over 2 Radar SPU
  - Antenna signal integration for elevation
  - Basic thresholding (local maximum detection)
- RAM size optimisation by using configurable FFT strategy (in place, out of place, partially shifted)
- Output DMA with output FIFOs so that each data flow has its own FIFO and configurable DMA parameters
- A min/max/averaging unit.
- A histogram unit up to 4096 elements
- Support for up to 8 data lanes @ 400Mbit/s
- Low speed 60 Mb/s single ended IF
- The ability to lockstep 2 SPU units.

### 19.2 Overview

This section provides an overview of the SPU functionality

#### 19.2.1 Glossary of Terms

The following terms are used throughout this document

**Table 591 Definition of Terms**

Term	Definition
Ramp	A frequency sweep of the RF transmitter of the Radar system. Relevant here because each ramp will result in the acquisition of a number of linked ADC samples to be processed by the SPU
Transposition	Reading input data from non-contiguous addresses in memory so as to construct an FFT input dataset from equivalent elements in multiple FFT results stored in memory. So, if the input data consists of 128 FFTs of 256 samples, the output data will transpose the coordinates to generate 256 FFTs of 128 samples
First Stage FFT	The first stage FFT processes ADC data to obtain range information
In Place FFT	SPU specific term used when the results of FFT calculations are stored in the memory locations cleared by reading the input data for the FFT calculations. As the input data is overwritten, the “In Place FFT” operation destroys the input data.
Second Stage FFT	The second stage (or doppler) FFT is assumed to use equivalent output bins from the first stage FFT of each ramp as input (see transposition). This allows the output from the second stage FFT to be used to determine velocity.
Third Stage FFT	The third stage FFT is assumed to use equivalent output bins from each antenna of the second stage FFT results
Chirp	Synonym for “Ramp”

## Signal Processing Unit (SPU)

**Table 591 Definition of Terms (cont'd)**

Term	Definition
Data Cube	Each set of data stored in memory has three indices that are used to uniquely define each data point. These can be considered as co-ordinates in a three dimensional system which leads to the term “data cube”. For raw ADC data, the indices are sample index, ramp number (corresponding to time) and antenna number. For processed data, the axes would be bin index, FFT number (corresponding to time) and antenna number.
Dataset	a single set of FFT data, either inputs or outputs
Data block	one or more datasets stored in buffer memory intended for processing by the SPU in a single operation. This can be considered as a two-dimensional array of data where one axis is bin index and the other is FFT index
DBF	Digital Beam Forming. Adding the equivalent bins from each antenna together to produce a single, combined FFT result. Data format is complex so that phase information is used in the addition
NCI	Non-Coherent Integration. Adding the power of each equivalent bin from each antenna together to produce a 1-dimensional array of power values

### 19.2.2 Processing Flow

The SPU is intended to perform three operations simultaneously

- Processing of Input Data into FFT datasets, combining one or more datasets into data blocks
- performing an FFT on each of the datasets
- Postprocessing the datasets and writing the results to Radar Memory

To do this the SPU uses four buffer memories. This imposes a hard limit on the amount of data that can be handled in a single operation. The buffer RAMs are organised in two pairs. The first pair stores input data for the FFT accelerator and the second pair stores the output data from the FFT accelerator. Therefore the FFT accelerator can work on one set of data while the Output Data Processor is working on the results from the previous set of data and the Input Data Processor is preparing the next set of data.

Each input data block contains enough samples from each attached antenna to perform one FFT for each antenna. This amount of data can be referred to as a ramp and the processing step is referred to as the “inner loop”

The SPU processes a programmable number of these data blocks for each trigger. This can be referred to as a measurement cycle and the overall processing operation is referred to as the “outer loop”. Each completed outer loop sequence generates one complete set of results (a data cube) in the Radar Memory.

All arithmetic operations in the SPU are fixed point unless explicitly stated otherwise.

### 19.2.3 Use Case examples

The Signal Processing Unit supports 3 main configurations that can be defined by software at any time.

- Configuration 1: 1st stage FFT
- Configuration 2: 2nd stage and 3rd stage FFT
- Configuration 3: mathematical operations only (without FFT)

Please note that some operations possible from logic point of view do not make sense (e.g. thresholding after 1st stage FFT). They are shown here to demonstrate the capabilities of the SPU.

## Signal Processing Unit (SPU)

The SPU is intended to provide an infrastructure for radar signal preprocessing. Its configurability allows support of different data processing flows. This is shown by most configuration registers not being directly related to Radar characteristics such as number of antenna or number of ramps.

Many of the basic computing steps are defined by the number of operands to read, the address offset for inner loop and address offset for outer loop.

This document defines limitations imposed by memory size to allow system designers to maximize the use of the SPU according to their application case.

The focus on providing a flexible computing infrastructure means that SPU does not have specific HW to check SPU configuration. The coherency of the SPU configuration should be checked by developers.

Examples of different configurations are shown in the following paragraphs.

### 19.2.3.1 SPU Configuration 1

The figure below is showing the different computation blocks for 1st stage FFT.

**Table 592 SPU Configuration 1**

	De-Interleaving	Math1	FFT Accelerator	Math2 (signal power)	Math2 (FFT Bins)	HW-DMA-Out
<b>Processing stage Config.</b>	Enabled	Enabled	Enabled	Disabled	Optional	Enabled
<b>Processing Options</b>	N/A	Window Coefficients Real/Complex	Real/Complex 16 / 32 bits Nr. of points	Signal Power Non-coherent Integration Histogram Local Max with thresholding	Optional Bin Range cutoff Complex Rescaling	DMA parameters per enabled output data flow

The SPU supports arbitrary removal of preselected BINs of the FFT results during processing by Math2 unit

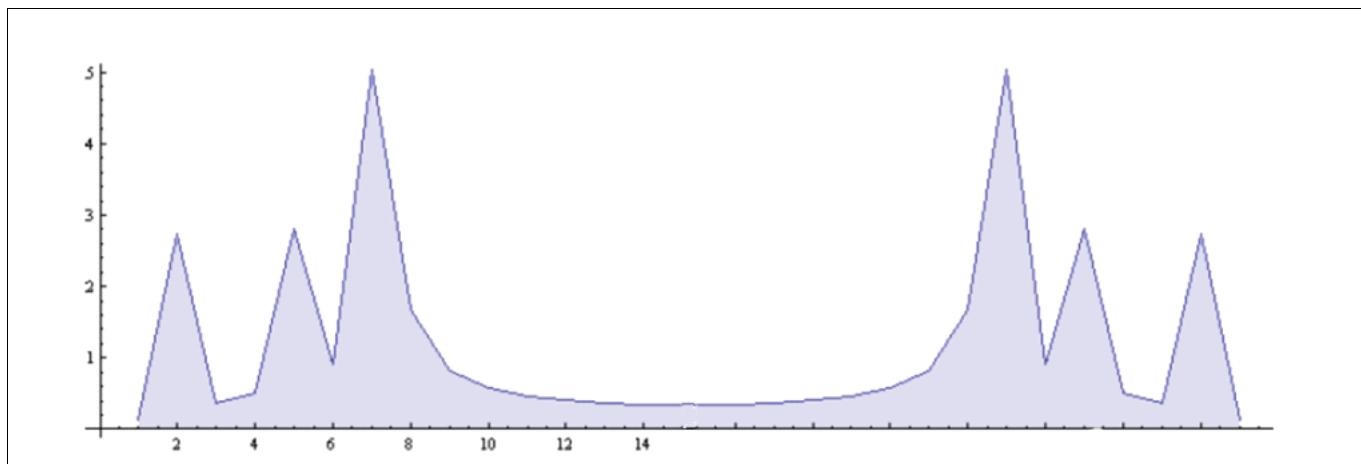
Ideally, the frequency sweep and other Radar parameters are chosen so that after the 1st stage FFT, only the BINs being symmetric with other ones are removed (e.g., keep the first 256 BINs from the 1st stage FFT512). Still, it might be necessary to remove some additional BINs.

- BINs from the real range to the equivalent range of the 256th BIN
- BINs on the very low range, which normally should have a 0 value

The FFT BIN removal HW in Math2 unit can be configured to remove any range BINs the user does not want to keep anymore.

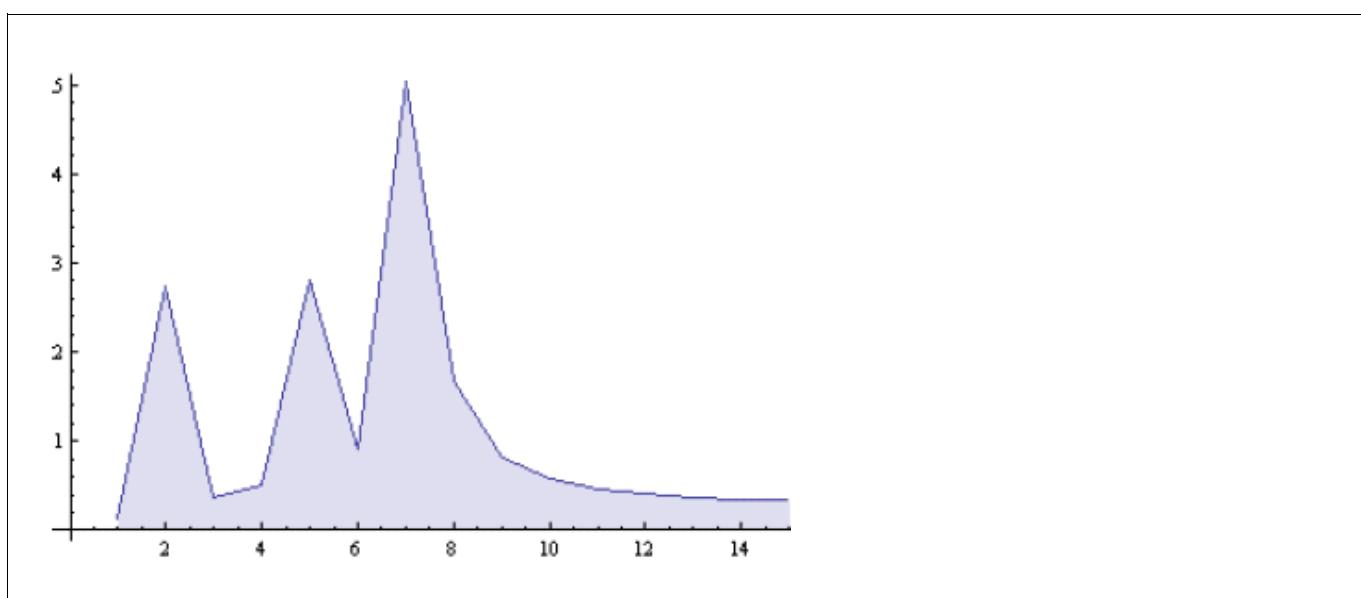
The symmetry of real FFT is illustrated in the figure below (FFT32 from 32 real samples).

## Signal Processing Unit (SPU)



**Figure 226 FFT Results Symmetry**

According to the above example, the range cut off would write to the Radar memory, only the first BINs (0 to 15) of the FFT results.



**Figure 227 Range Cut Off Example**

The Arbitrary BIN removal also allows additional optimization like removing the BIN0 and BIN1, so that only BINs outside the minimum range are stored in memory.

The Arbitrary BIN removal also allows additional optimization like removing the BIN13, BIN14 and BIN15 if the sampling of the Radar (frequency sweep, sampling time,..) is such that BIN13, BIN14 and BIN15 are not meaningful.

### 19.2.3.2 SPU Configuration 2

The main focus of this configuration is performing 2nd stage FFT, 3rd stage FFT and non coherent integration of the data from multiple antennae.

## Signal Processing Unit (SPU)

**Table 593 SPU Configuration 2**

	De-Interleaving	Math1	FFT Accelerator	Math2 (signal power)	Math2 (FFT Bins)	HW-DMA-Out
<b>Processing stage Config.</b>	Enabled. HW Transpose	Enabled	Enabled	Optional	Optional	Enabled
<b>Processing Options</b>	DMA Inner Loop Parameters DMA Outer Loop Parameters	Window Coefficients Real/Complex	Real/Complex 16 / 32 bits Nr. of points	Signal Power Non-coherent Integration Histogram Local Max with thresholding	Bin rejection / patch based on thresholding Complex Rescaling Bin Forcing to 0 Min / Max / Averaging Sum / Sum-abs. Sum-ant	DMA parameters per enabled output data flow (allows in-place FFT)

Comments on configuration2

- Non-coherent integration is done within 1 SPU, even when using 2 SPUs. Integration across two SPU instances is not supported.

### 19.2.3.3 SPU Configuration 3

The figure below shows the different computation blocks for the 3rd configuration.

Configuration 3 is used to do specific computations on FFT BINs or on the result of non coherent integration of teh data from multiple antennae

Configuration 3 is useful to do specific computations before or after FFTs like

- FFT result multiplication by another complex (scalar or vector)
- maximum search in the other dimension to the one used with configuration2
- complex thresholding algorithm like CFAR
- average computation

This dedicated stage has been specifically designed to allow the running of the 2nd stage FFT at full performance of the Radar SPU (so, at 2 clocks / BIN) and produce then additional and specific results at 4 clocks /BIN during the time the CPU is processing the results produced by the 2nd stage FFT.

This provides more flexibility in supporting computation flows.

## Signal Processing Unit (SPU)

**Table 594 SPU Configuration 3**

	De-Interleaving	Math1	FFT Accelerator	Math2 (signal power)	Math2 (FFT Bins)	HW-DMA-Out
<b>Processing stage Config.</b>	Enabled. HW Transpose	Optional	Disabled	Optional	Optional	Enabled
<b>Processing Options</b>	DMA Inner Loop Parameters DMA Outer Loop Parameters	Window Coefficients Real/Complex	Real/Complex 16 / 32 bits Nr. of points	Signal Power Non-coherent Integration Histogram Local Max with thresholding	Bin rejection / patch based on thresholding Rescaling Bin Forcing to 0 Min / Max / Averaging Sum / Sum-abs. Sum-ant	DMA parameters per enabled output data flow (allows in-place FFT)

Note: Comments on configuration 3

4. 2 independent thresholding algorithms are computed concurrently. Users can then select which of the thresholding results to use on any results based on their own software algorithm run on the TriCore.

### 19.2.3.4 Thresholding

The SPU provides hardware support for different thresholding methods

- user defined thresholding
- local min/max
- CFAR based thresholding methods

#### 19.2.3.4.1 User Defined Thresholding

The SPU is able to compute both a histogram and statistical data of the signal power over a selected region of the data (entire data cube, partial data cube, cube slice,...).

Once the histogram and statistical data are produced by the SPU, the CPU can then process it to compute the threshold to be applied for the next computation step.

For this, the CPU has to update the relevant entry in the radar sequencer with the suitable thresholding parameter and enable the Radar sequencer to load the next configuration and execute it.

#### 19.2.3.4.2 CFAR Based Thresholding Methods

CFAR algorithms can be quite complex and CPU time demanding. The CFAR module in the MATH2 unit can run two, CFAR algorithms concurrently and store their results in memory so that users can select, based on their own methodology, which results they apply for each part of the radar image.

### 19.2.3.5 Using Pre-acquisition Ramps

Running thresholding on FFT results from 1 ramp does not usually allow enough gain.

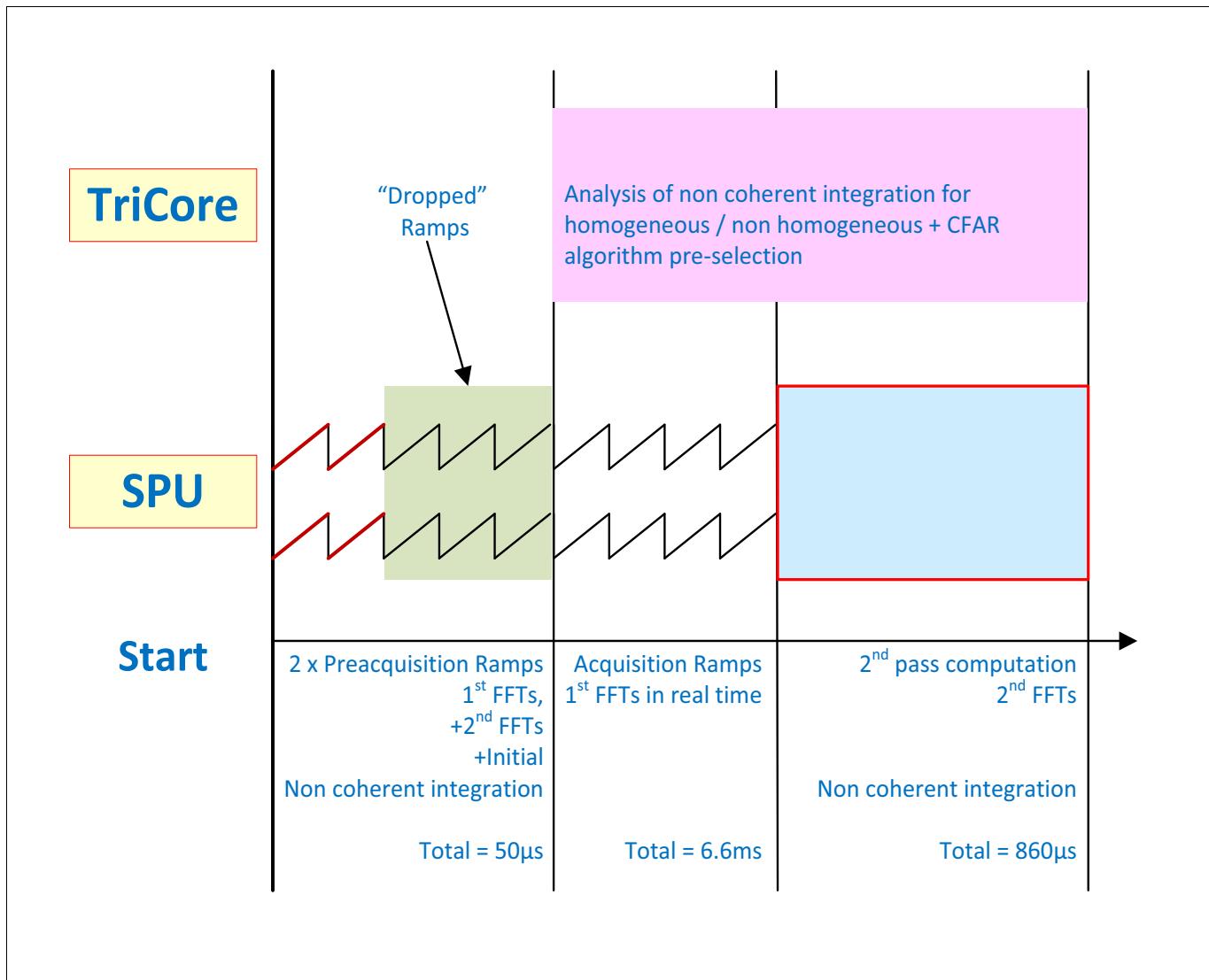
## Signal Processing Unit (SPU)

The SPU supports running a set of pre-acquisitions ramps to allow to an initial non coherent integration and initial thresholding to be performed.

Then, after few dummy ramps, regular acquisition ramps can start.

Such an approach allows the CPU to run a user defined SW to analyse an initial non coherent integration and initial thresholding of the pre-acquisition data to understand the complexity of the scenery.

This is illustrated in the figure below



**Figure 228 Pre-Acquisition Ramps Method**

This technique is not possible with slow ramps but can be used with fast ramps.

**Note:** pre-acquisition ramps could be seen as partial redundant acquisition so that a high integrity decision can be taken from 1 acquisition only.

### 19.2.4 Elevation support

Support of elevation computations is anticipated to become necessary as, when a car makes emergency brake decisions based on Radar, it is important to detect if a potential target is on the driving path or just above it.

Computation of elevation may require summing the signals from antenna pairs.

## Signal Processing Unit (SPU)

**Patents on antenna topologies:** Some antenna topologies have been patented. The Radar SPU is only providing an infrastructure for computations. It is the end user responsibility to make sure their system/application usage of the Radar SPU is not violating system or application level patents.

**Note:** *The Math1 supports automatic padding so that the 3rd stage FFT can be run automatically as an additional computing step after the sum of antenna pairs on 2nd stage FFT results is computed.*

### 19.2.5 Phase demodulation

2 methods are supported inside the Math1 unit.

- alternate chirp demodulation
- static demodulation

Principle: phase demodulation is done by performing a complex multiplication on input data by re-using the windowing function. i.e. the required phase demodulation is incorporated into the co-efficients of the windowing function.

**Note:** *as the demodulation scheme uses complex multiplication, the SPU supports demodulation of phases other than 180 degrees.*

#### 19.2.5.1 Alternate chirp demodulation

When doing demodulation per chirp, it is assumed that data are modulated on a chirp by chirp basis (odd/ even).

The SPU allows demodulation prior to 1st stage FFT or prior to 2nd stage FFT.

The output-DMA supports writing each flow of FFTs to different addresses. This allows the Radar SPU to run further FFTs over all ramps or over 1 set of ramps (odd or even ramp indexes) and, as part of the output-DMA features, to define where the FFT results are stored.

User can select their own method for doing phase demodulation as SPU support different implementations.

#### 19.2.5.2 Static demodulation

When doing static demodulation, the SPU provides resources to select between 2 sets of windowing coefficients.

#### 19.2.5.3 HW optimisation for demodulation

Demodulation using the windowing function in Math1 defines constraints on the number of active antenna and the number of bins in each FFT as the space available for storing window functions in the Configuration Memory is limited.

When using 0/90/180/270 phase demodulation, fast demodulation allows demodulation without using the windowing function and therefore without using Configuration Memory.

### 19.2.6 Debugging

For debugging purposes, it is possible to stop the execution of a sequence at a specific FFT execution step (e.g. 1st stage or 2nd stage). This is done by the radar sequencer.

To avoid excessive complexity, execution cannot be resumed.

Debugging is supported by

- the connections to the OCDS system (See [Section 19.5, “Debug”](#))
- the defined interrupt flags.
- the counters added to support functional safety.

## Signal Processing Unit (SPU)

### 19.2.7 Execution flow

The execution flow shown here shows the mapping of 1 FFT engine. Execution flow of the 2nd FFT engine is identical, only the base address is shifted.

#### 19.2.7.1 Execution flow for 4 Antennae

Use case shown is for a first stage FFT with 4 antennae. The Radar Interface will receive one set of sample data for each attached antenna per ramp of the acquisition cycle. In the example shown in [Table 595 “Execution Flow with 4 Antennae” on Page 10](#), 512 ramps are assumed. Since the buffer memories are relatively small, the SPU is constrained to process the data in order of acquisition. The SPU will iterate round the inner loop once for each antenna for each iteration of the outer loop. The number of outer loop iterations will match the number of ramps in the acquisition cycle.

It therefore follows that the amount of data generated by one ramp from each antenna must fit into the buffer memory. In all cases, ADC data is stored as sixteen bit quantities.

**Table 595 Execution Flow with 4 Antennae**

Outer Processing Loop (increasing time)	Inner Processing Loop (Increasing Time ->)			
1	FFT0, Antenna 0	FFT 0, Antenna 1	FFT 0, Antenna 2	FFT 0, Antenna 3
2	FFT1, Antenna 0	FFT 1, Antenna 1	FFT 1, Antenna 2	FFT 1, Antenna 3
3	FFT2, Antenna 0	FFT 2, Antenna 1	FFT 2, Antenna 2	FFT 2, Antenna 3
4	FFT3, Antenna 0	FFT 3, Antenna 1	FFT 3, Antenna 2	FFT 3, Antenna 3
5	FFT4, Antenna 0	FFT 4, Antenna 1	FFT 4, Antenna 2	FFT 4, Antenna 3
6	FFT5, Antenna 0	FFT 5, Antenna 1	FFT 5, Antenna 2	FFT 5, Antenna 3
7	FFT6, Antenna 0	FFT 6, Antenna 1	FFT 6, Antenna 2	FFT 6, Antenna 3
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
509	FFT508, Antenna 0	FFT 508, Antenna 1	FFT 508, Antenna 2	FFT 508, Antenna 3
510	FFT509, Antenna 0	FFT 509, Antenna 1	FFT 509, Antenna 2	FFT 509, Antenna 3
511	FFT510, Antenna 0	FFT 510, Antenna 1	FFT 510, Antenna 2	FFT 510, Antenna 3
512	FFT511, Antenna 0	FFT 511, Antenna 1	FFT 511, Antenna 2	FFT 511, Antenna 3

*Note:* In the case where only 3 antennae are used, Antenna 3 would be disabled.

### 19.2.8 Memory mapping

This section provides an overview with example use cases of the configuration options available for the input and output DMA engines.

## Signal Processing Unit (SPU)

### 19.2.8.1 Principle

The Input DMA engine is designed to do transfers with programmable address increments. Three separate increments are available which are combined with a base address to build the address to be accessed. This allows a DMA transfer to be configured which assembles elements from a 3-dimensional array (or data cube) stored as a flat (1-dimensional) array in Radar Memory by stepping along any axis in the cube with any arbitrary starting offset.

These increments are used in a “bin loop”, an “inner loop” and an “outer loop” processing flow. The inner loop can potentially be executed multiple times for each execution of the outer loop and the bin loop can be executed multiple times for each execution of the inner loop.

Processing starts with a pre-defined base address and a zero offset address for each loop. These are added together to form the access address. Each pass round the bin loop increments the bin loop offset, each pass round the inner loop will increment the inner loop offset address and finally each pass round the outer loop increments the outer loop offset. The increment for each offset register is programmable. When the programmed number of loops have completed, the outer loop restarts by resetting the offset address and loop counter to zero.

A complete execution of the “bin loop” will read the data for a single FFT input dataset. A complete execution of the “inner loop” would therefore be expected to create a data block but, in a lot of cases, this would create a data block too large to fit into the buffer memory. Smaller data blocks are therefore created. See [Section 19.2.8.3, “Data Block Construction Control” on Page 11](#) for details.

The Output DMA engine is much simpler. Data is written to sequential addresses in order of processing unless in-place FFT is enabled in which case, the output address will be calculated to write the data into the addresses made available by the reading of the input data.

### 19.2.8.2 Memory mapping for FFTs

FFTs can be arranged to be done “in-place” or “out-of-place”. When done “out-of-place”, a different memory space is used for input and output data. This can either be completely segregated (necessary for transposed addressing on the input data) or overlapping with a shift in base address (1 or few ramp differences/address offset).

Having a difference of only a few ramp allows some of the 1st stage FFT results to be retained in the Radar Memory after the 2nd stage processing is complete.

When using in-place operation with transposed addressing of the input data. A scatter algorithm will be used when writing data which is the inverse<sup>1)</sup> of the gather algorithm used for reading data.

### 19.2.8.3 Data Block Construction Control

The Input DMA constructs a data block in buffer memory consisting of multiple datasets. The point at which the data block is defined as complete and the buffer memory is switched so that the data can be processed can be configured in one of two modes:

#### 19.2.8.3.1 Default Memory Mode

This mode is used when there is no requirement to use either coherent or non-coherent integration across the FFT datasets stored in buffer memory. This allows maximum flexibility on switching the buffer memory during the execution of the inner loop.

<sup>1)</sup> “inverse” as the gather operation takes scattered input data and generates contiguous output data and the scatter operation takes contiguous input data and generates scattered output data

## Signal Processing Unit (SPU)

### 19.2.8.3.2 Integration Mode

This mode is used when there is a requirement to use either coherent or non-coherent integration across the FFT datasets in each data block. The normal use case anticipated for coherent or non-coherent integration is for combining results from multiple antennae into a single set of data.

For this to be useful, the data block constructed on the buffer memory must then contain one dataset for each value of the Inner Loop Count or the integration results will not be correct. In this case ID\_RM\_CONF.PM should be set to IM (Integration Mode). This forces the switching of the buffer memory to occur at the end of each iteration of the inner loop.

Provided that there is no address transposition (i.e. the datasets constructed in buffer memory are copies of data sets in radar memory, this makes efficient use of Radar Memory bandwidth as a single data block uses all of each 256 bit word read from Radar memory. If there is address transposition, then the bandwidth optimisation described below can be used

### 19.2.8.4 Bandwidth Optimised Integration Mode

Bandwidth Optimised Integration mode of the Input DMA can be used to optimise Radar Memory bandwidth usage when coherent or non-coherent integration across the antennae is required by the MATH2 Unit at the same time as using transposed addressing (i.e. building an FFT dataset from the same sample bin in multiple FFT results in Radar Memory).

The objective is to build an input data block containing one FFT dataset for each antenna. This can then be processed by the MATH2 streaming processor to generate the integration output.

For the four antenna use case, with 256 FFTs or ramps in the input data and 128 bins per FFT or ramp, the objective is to sequentially create the data blocks

- Data Block 0 (A{0,0:1,0:2,0:...:254,0:255,0},B{0,0:1,0:...:255,0},C{0,0:1,0:...:255,0},D{0,0:1,0:...:255,0})
- Data Block 1 (A{0,1: ... :255,1},B{0,1: ... :255,1},C{0,1: ... :255,1},D{0,1: ... :255,1})
- .
- .
- .
- Data Block 126 (A{0,126: ... :255,126},B{0,126: ... :255,126},C{0,126: ... :255,126},D{0,126: ... :255,126}),
- Data Block 127 (A{0,127: ... :255,127},B{0,127: ... :255,127},C{0,127: ... :255,127},D{0,127: ... :255,127})

where **A{n,m:n,m:...:n,m}** indicates an input data array assembled from antenna **A**. **{..}** is a colon separated array of datapoints where each datapoint is described by two numbers, **n** and **m**, where **n** is the FFT and **m** is the bin.

The issue addressed by bandwidth optimisation is that the first word read from memory will contain A{0,0:0,1:0,2:0,3}. This results in 75% of the data read being discarded (for 32 bit precision data) as A{0,0} is needed for data block 0; A{0,1} is needed for data block 1; A{0,2} is needed for data block 2 and A{0,3} is needed for data block 3.

Bandwidth optimisation allows multiple datablocks to be constructed simultaneously in the input buffer memory thus using all the data read from the Radar Memory. These datablocks are then passed sequentially through the streaming processors. This bandwidth optimisation is done by effectively executing multiple iterations of the outer loop simultaneously.

This mode can only work if the multiple data blocks can be fitted into the available space of the Buffer RAM. If this is not the case, Integration mode cannot be enabled and the input DMA will iterate over the data multiple times to sequentially build the data blocks one at a time.

## Signal Processing Unit (SPU)

Bandwidth optimisation of Integration mode is enabled by setting ID\_RM\_CONF.PM to IM (Integration Mode) and ID\_RM\_CONF.BLOCKS to a value other than 0<sub>D</sub>. The ID\_RM\_CONF.BLOCKS bitfield then controls the number of datablocks to be built.

See [Section 19.3.1.2.6, Bandwidth Optimisation Integration Processing Mode.](#), for information on configuring this mode.

### 19.2.8.5 Memory mapping for other SPU results

Other SPU results (non coherent integration, CFAR outputs) will be stored in a dedicated memory region.

### 19.2.8.6 Data sharing between SPU

Each SPU should have exclusive access to its memories. The Radar memory being organized in tiles, tiles can be statically allocated to any of the SPUs. The dependencies between the 2 x SPU (when 2nd SPU is implemented) is only visible to users for the 3rd stage FFTs and for support of elevation computations.

#### 19.2.8.6.1 Complex memory map via DMA reconfiguration

Because of the pipeline on acquisition, DMA reconfiguration during acquisitions is possible but will require use of the linked list configuration option.

FFT results of partial-acquisition ramps can be stored in separated memory tiles so that the CPU core(s) can read the results of partial-acquisition ramps during the time of acquisitions and FFTs on the main acquisition ramps.

This also applies for post /pre acquisition test data coming from the RF device.

As the memory mapping is configurable, the rest of the description of the memory mapping should be taken as examples rather than an exhaustive description of what is possible.

The memory mapping examples show the mapping of 1 SPU. Memory mapping of the 2nd FFT engine is identical; only the base address is shifted.

The writing strategy for FFT results is that FFT results for a single FFT are written in sequential memory locations. The start address for each FFT result set is 32 byte aligned to aid in efficient memory usage. One set of FFT results from each antenna will be grouped together in a single ramp result set.

The writing strategy for 2nd stage FFT results is to write the data in order of processing with each FFT result grouped in contiguous addresses and starting at a 32 byte aligned address. This holds unless in-place storage of the FFT results is required (“in place FFT”). In this case, data will be written to memory using the inverse of the transpose algorithm used to build the input data sets and the results will therefore be stored in non-linear addresses.

### 19.2.8.7 Example Memory mapping for 4 antenna and 16 bit operands

The figure below is showing the content of the 256bit words in Radar memory after 1st stage FFT. The example is for a 256 ramp acquisition cycle with four antennae storing 128, sixteen bit complex results (i.e. a 32 bit word representing a complex number with 16 bit signed integers for real and complex components). This results in eight being stored in each 256 bit word of Radar Memory.

## Signal Processing Unit (SPU)

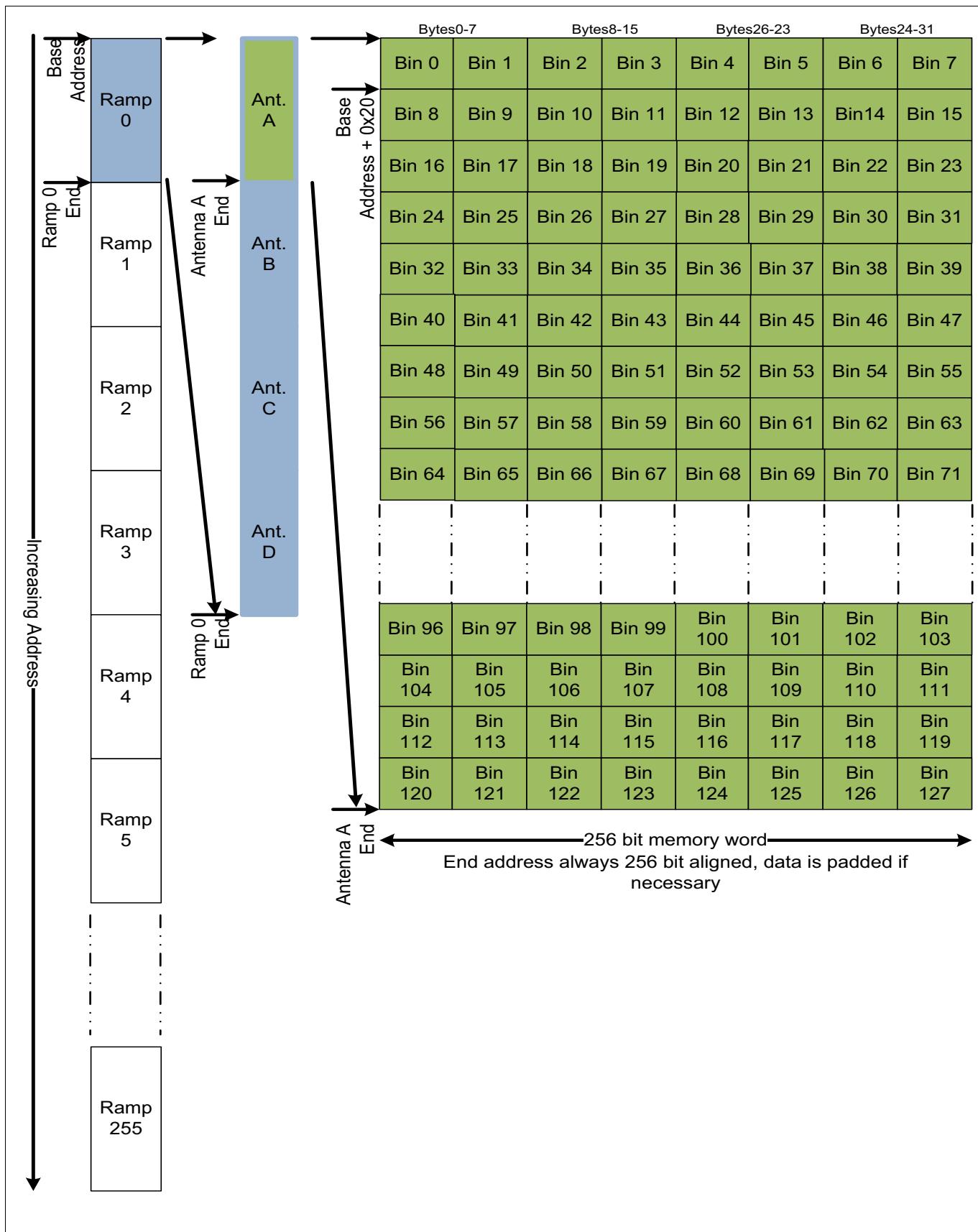


Figure 229 First Stage FFT Memory Organisation, 16 bit data, 4 antennae

## Signal Processing Unit (SPU)

### 19.2.8.8 Data Read Order for 2nd stage FFT with 4 antennae and 16 bit operands

The figure below show the default Input DMA Unit read sequence for transpose before 2nd stage FFTs with 16bit operands and 4 antenna. Here the data is labelled as  $X_{n,m}$ , where X is the antenna identifier, n, is the ramp number and m is the bin number. There is one bin loop for each execution of the inner loop.

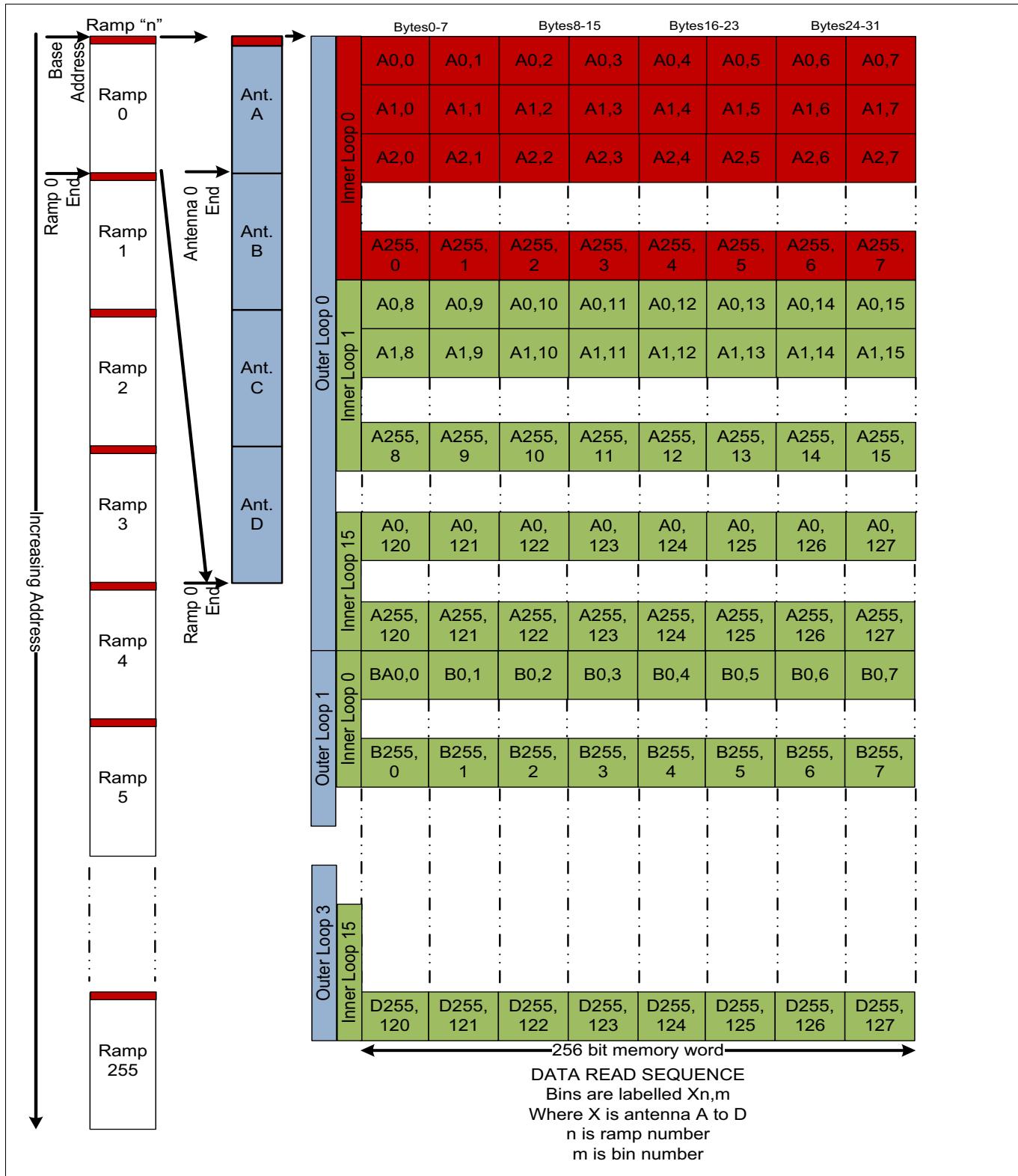


Figure 230 Transpose read order for 2nd stage FFT, 16 bit data, 4 antennae

## Signal Processing Unit (SPU)

For each 256 bit word, we read 8 bins of 1 antenna and go through the same antenna in the 2nd dimension of the datacube.

8x FIFOs are required in the transpose DMA for 16 bit data. This results in eight sets of input data being constructed in the buffer memory one for bin(m) from each ramp, one for bin(m+1), one for bin(m+2) and so on up to bin(m+7), where m is always divisible by eight.

In this case, the inner loop cycles over bins 0 to m(max) where m(max) is the number of bins in the input data divided by 8 and then the outer loop cycles over the antennae.

### 19.2.8.8.1 Data Mapping for 2nd Stage FFT results with 16bit Operands and 4 Antennae

The figure below is showing Radar memory for 16bit operands and 4 antenna and shows 256bit word content after 2nd stage FFT. This assumes no “in place FFT” with transposed addressing.

## Signal Processing Unit (SPU)

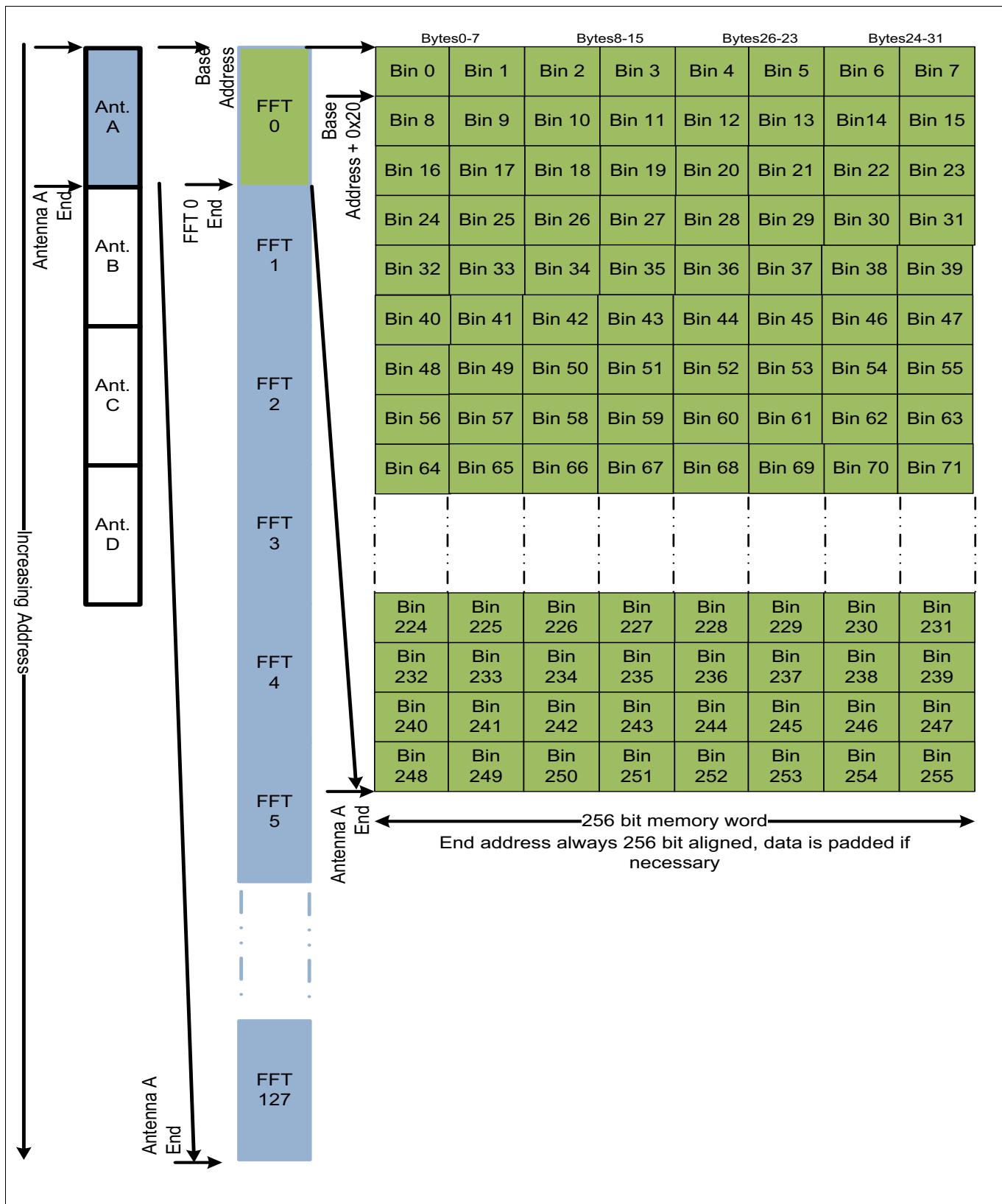


Figure 231 Second Stage FFT Memory Organisation, 4 antennae with 16 bit operands

### 19.2.8.8.2 Data Read Sequence in Integration Mode

The following figure shows the Input DMA Unit read sequence for 16 bit data and 4 antennae when coherent or non-coherent integration across the antennae is required. This shows individual reads and is not optimised for

## Signal Processing Unit (SPU)

bandwidth. See [Chapter 19.2.8.4, Bandwidth Optimised Integration Mode](#) for details on bandwidth optimisation.

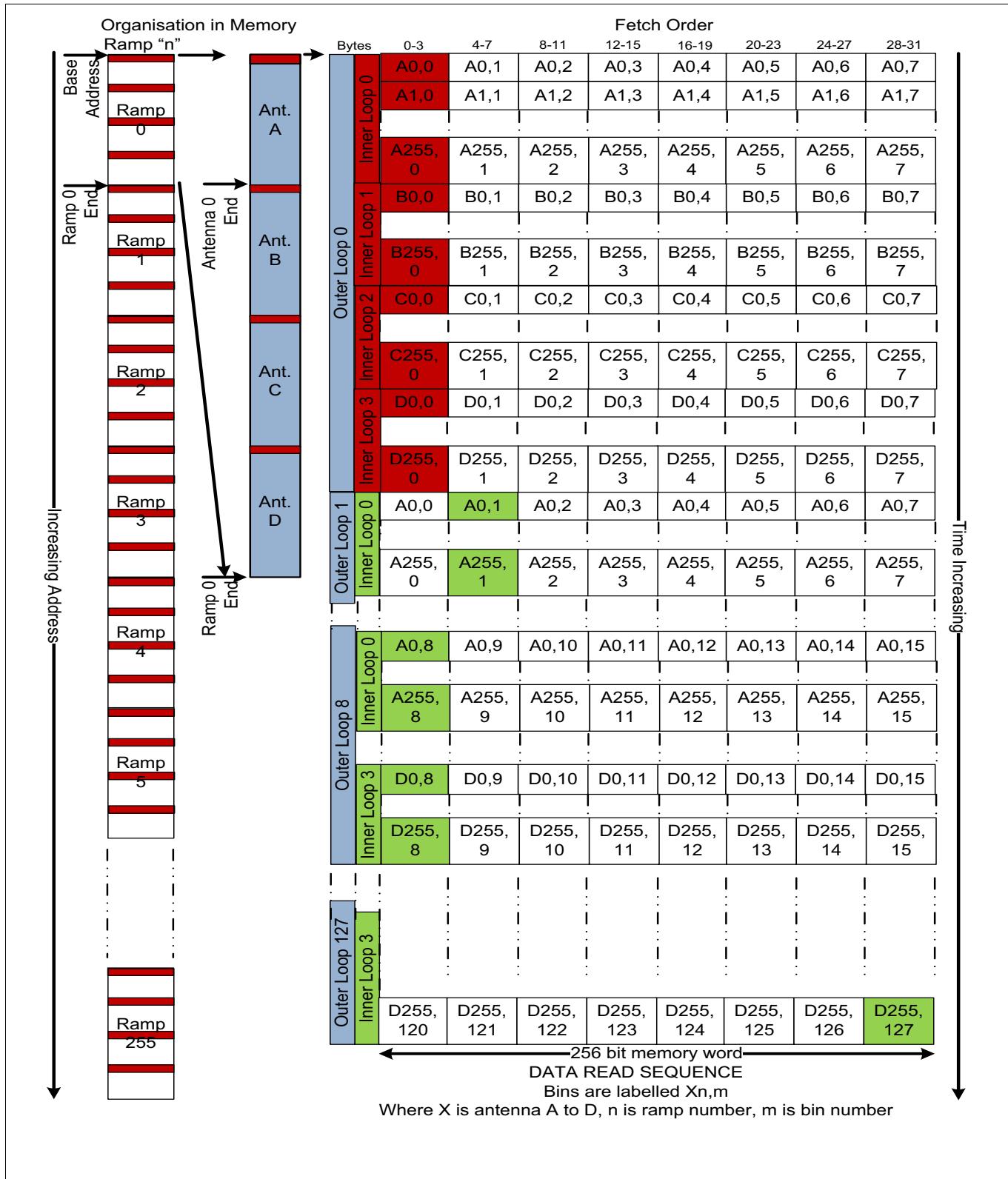


Figure 232 Integration Mode Transpose read order for 2nd stage FFT, 16 bit data, 4 antennae

### 19.2.8.8.3 Data Mapping for Integration Mode 2nd Stage FFT results with 16bit

## Signal Processing Unit (SPU)

### Operands and 4 Antennae

The figure below is showing Radar memory for 16 bit operands and 4 antennae. Showing 256 bit word content after 2nd stage FFT. This assumes no “in place FFT” with transposed addressing.

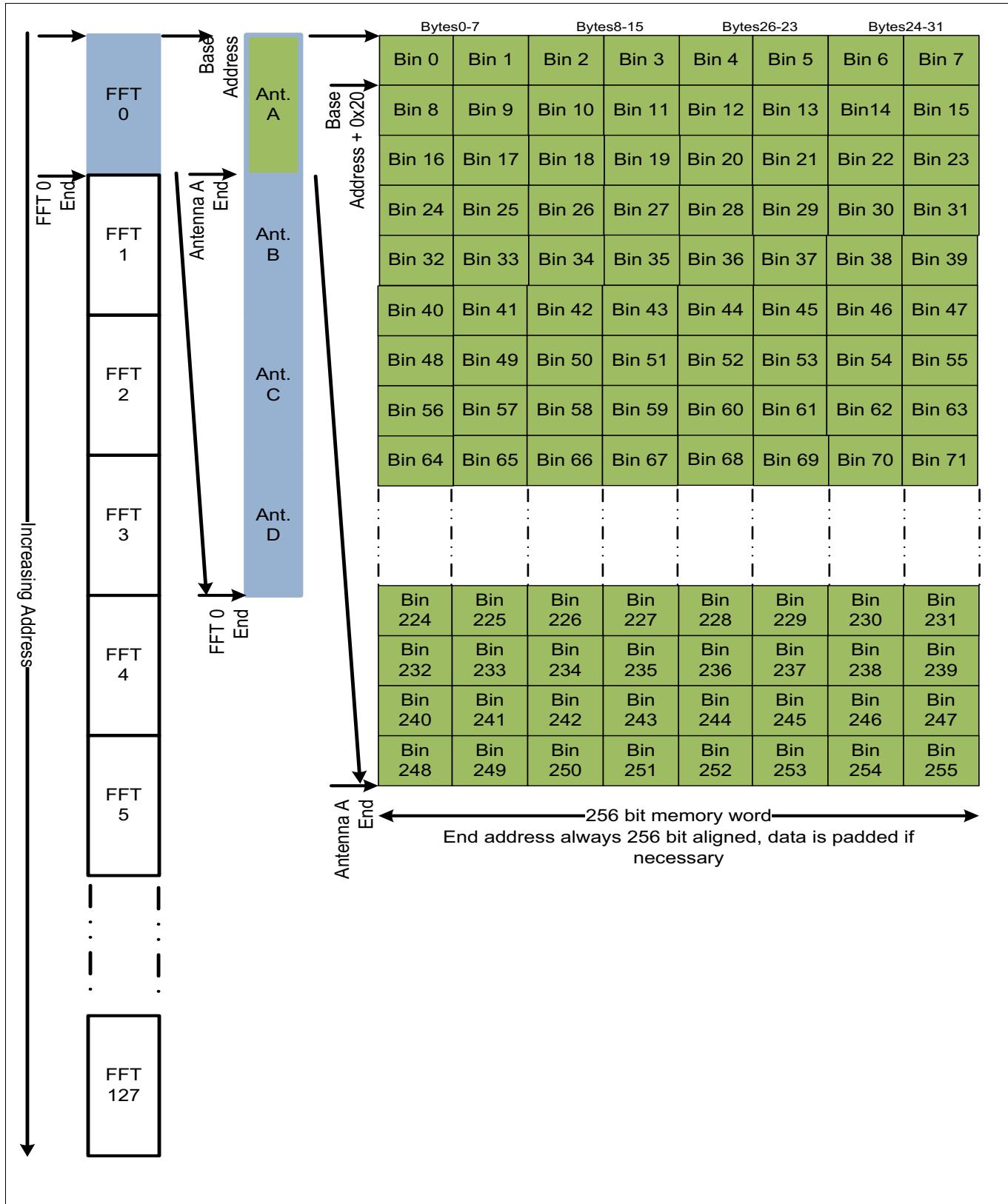


Figure 233 Integration Mode, 2nd Stage FFT Data Organisation, 4 antennae with 16 bit operands

---

## Signal Processing Unit (SPU)

### 19.2.8.9 3rd stage FFT

Ability to run systematic 3rd stage FFTs is of interest for safety reason.

Ability of running systematic 3rd stage FFT might become mandatory for elevation support: during the time the CPU will run super resolution algorithm on DOA, the Radar SPU can compute low resolution angle of elevation. the Radar SPU, as it is defined here, is able to perform 3rd stage FFT

- operands alignment in memory is prepared for this (can use same DMA read sequence)
- Input DMA Unit supports “zero” padding on its input.

#### Notes: Comment on safety aspects

1. *when running complex angle estimation algorithms like MUSIC, ESPRIT, ROOT-MUSIC, you will then have an estimate of the angle position.*
2. *this estimate should match the gross estimate provided by the systematic 3rd stage FFT.*
3. *having them done by the accelerators allows to have the TriCore fully available to run the precise angle estimation algorithm.*
4. *So, here performance is not so much the focus.*

## Signal Processing Unit (SPU)

### 19.2.8.10 Memory mapping for 4 antenna and 32bit operands

The figure below is showing the content of the 256 bit words in Radar memory after 1st stage FFT.

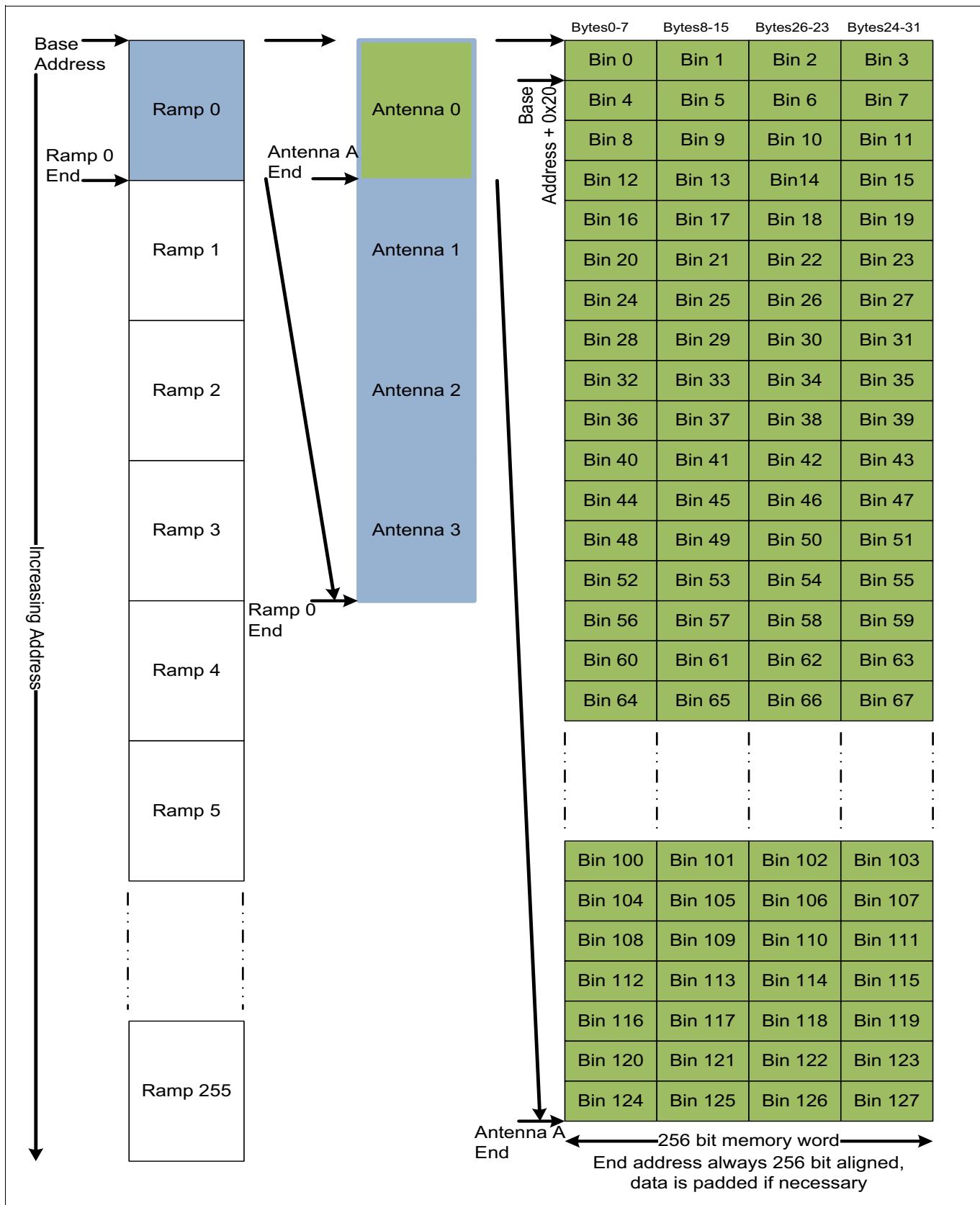


Figure 234 1st stage FFT Memory Organisation, 32 bit data, 4 antennae

## Signal Processing Unit (SPU)

### 19.2.8.11 Data Mapping for 2nd Stage FFT with 4 antennae and 32 bit operands

The figure below is showing Input DMA Unit read sequence for transpose before 2nd stage FFTs with 32bit operands and 4 antenna

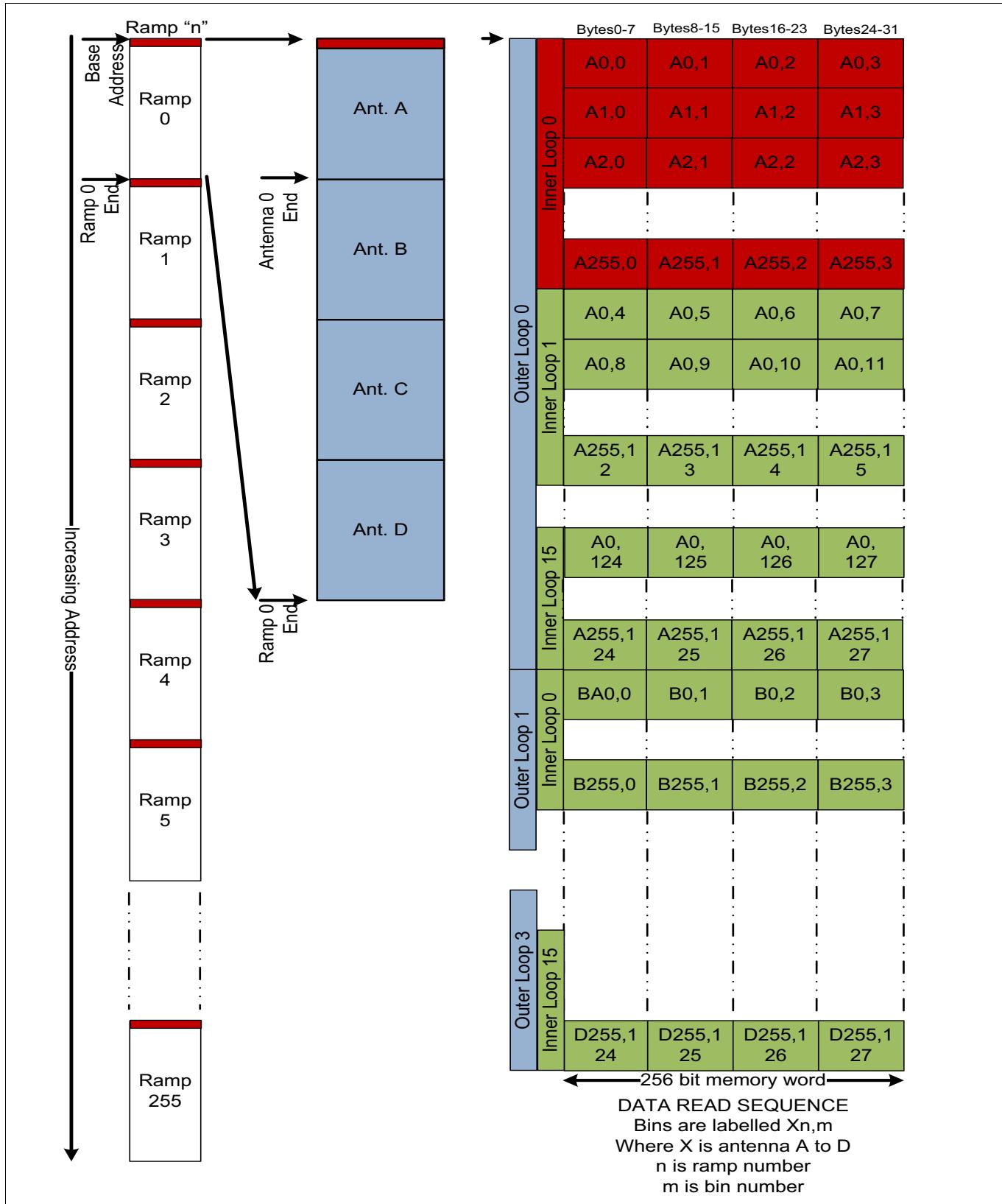


Figure 235 Transpose read order for 2nd stage FFT, 32 bit data, 4 antennae

## Signal Processing Unit (SPU)

The figure below is showing Radar memory for 32bit operands and 4 antenna and shows 256 bit word content after 2nd stage FFT. This assumes no “in place FFT” with transposed addressing.

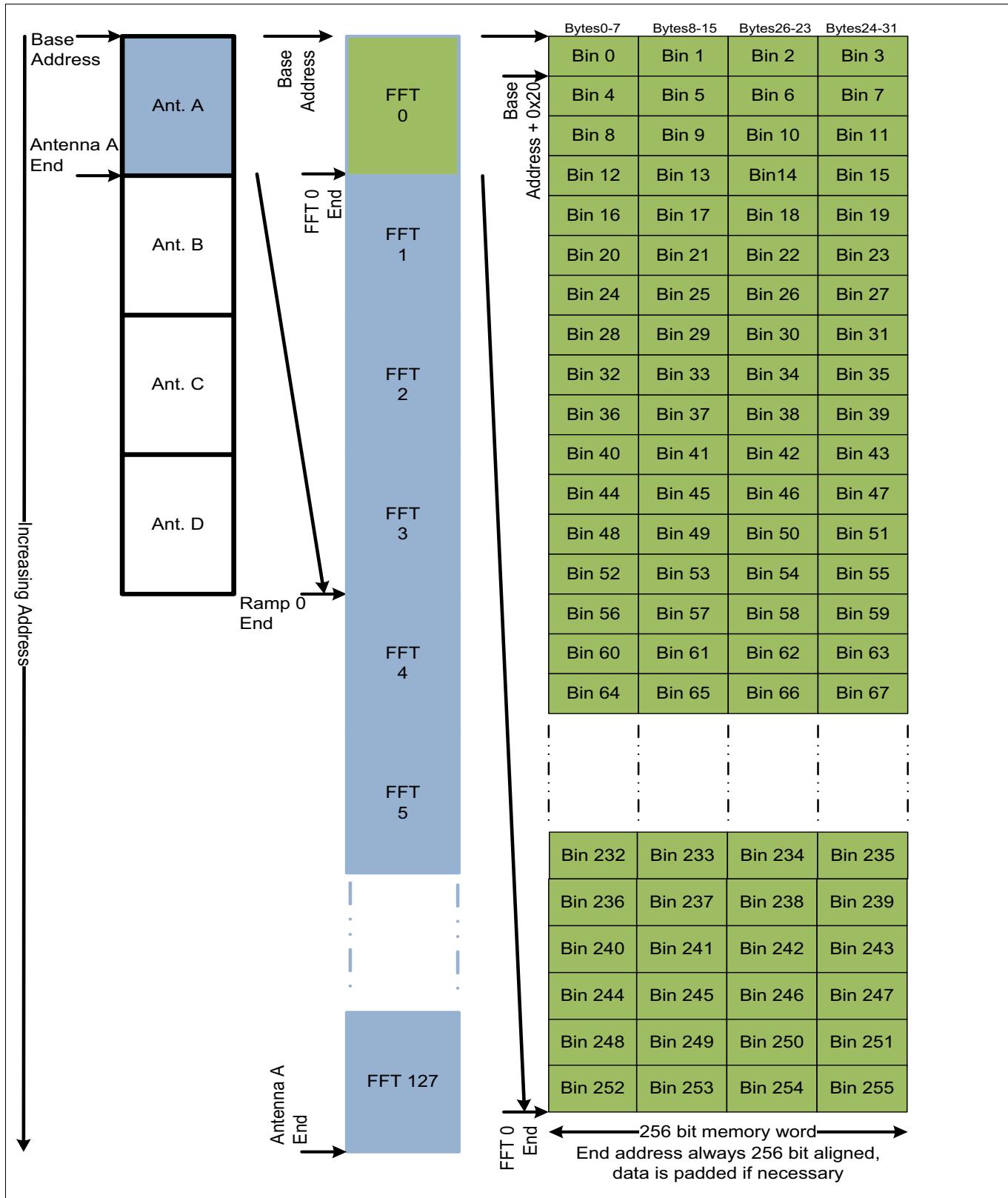


Figure 236 2nd Stage FFT Data Organisation, 4 antennae with 32 bit operands

## Signal Processing Unit (SPU)

### 19.2.8.12 Data Organisation in Integration Mode

The following figure shows the Input DMA Unit read sequence for 32 bit data and 4 antennae when coherent or non-coherent integration across the antennae is required. This shows individual reads and is not optimised for bandwidth. See [Chapter 19.2.8.4, Bandwidth Optimised Integration Mode](#) for details on bandwidth optimisation.

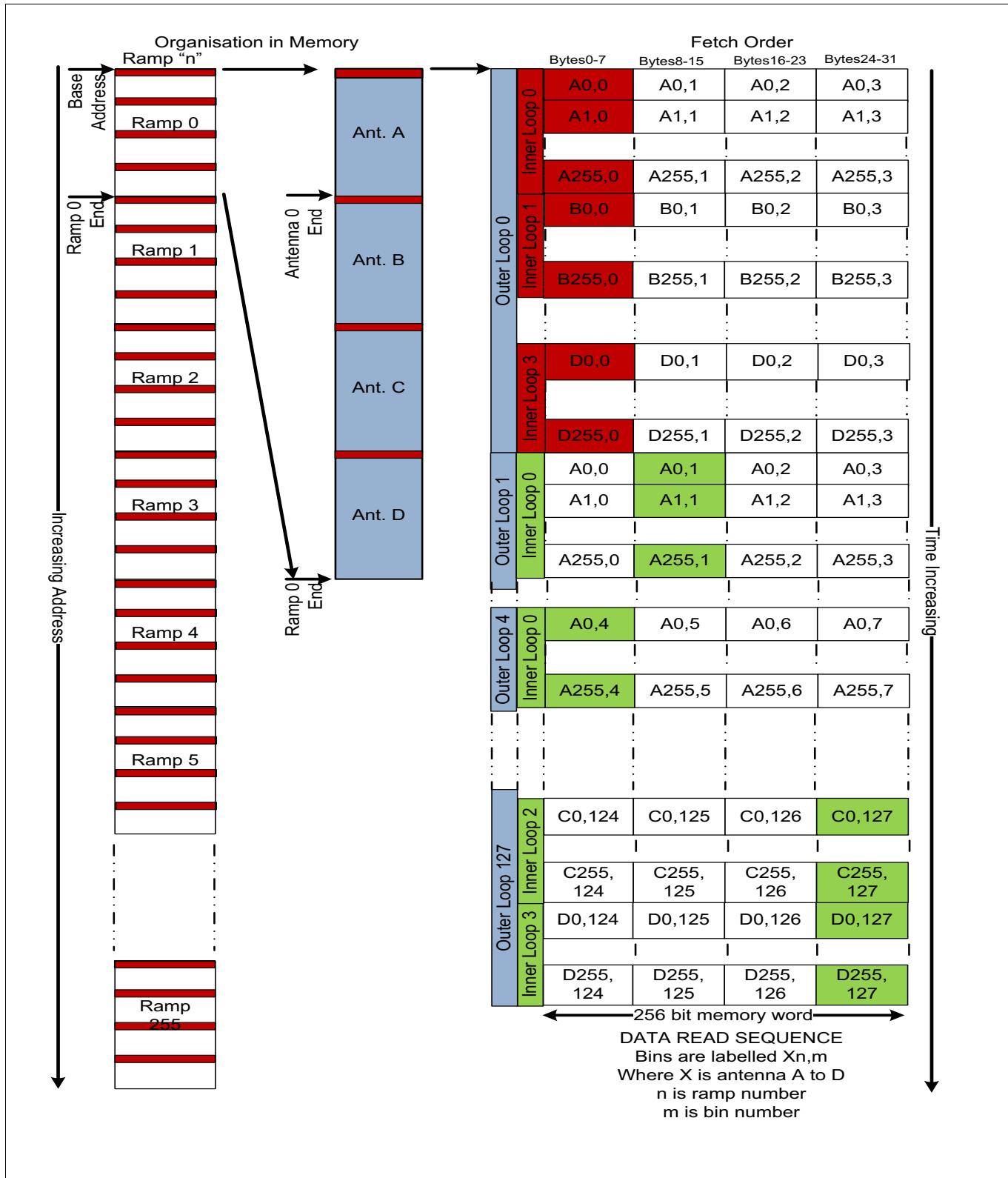


Figure 237 Integration Mode Transpose read order for 2nd stage FFT, 32 bit data, 4 antennae

## Signal Processing Unit (SPU)

The figure below is showing Radar memory for 32bit operands and 4 antennae. Showing 256 bit word content after 2nd stage FFT. This assumes no in place FFT with transposed addressing.

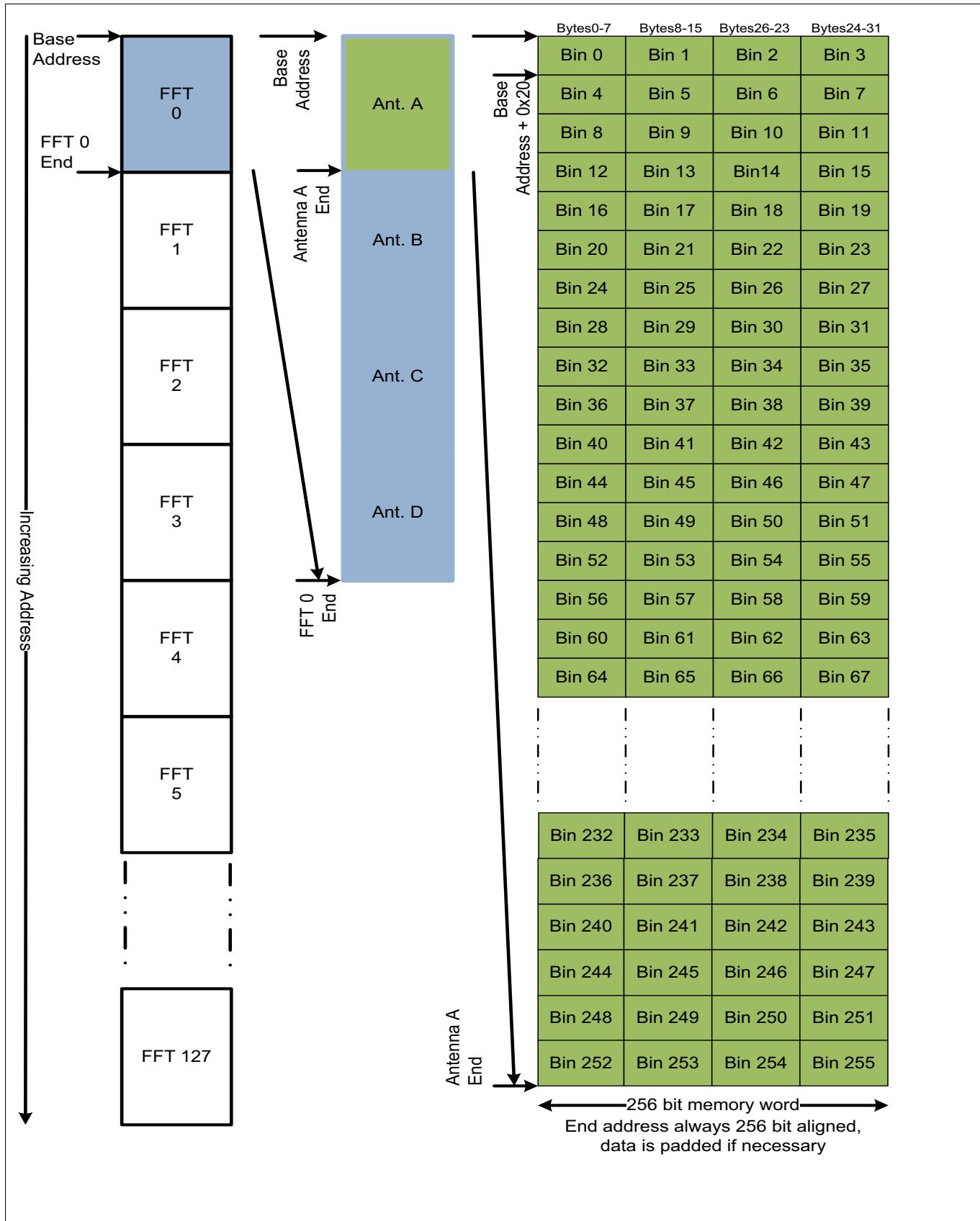


Figure 238 Integration Mode, 2nd Stage FFT Data Organisation, 4 antennae with 32 bit operands

## Signal Processing Unit (SPU)

### 19.2.8.13 Memory mapping for 3 antenna and 16bit operands

The figure below is showing the content of the 256 bit words in Radar memory after 1st stage FFT.

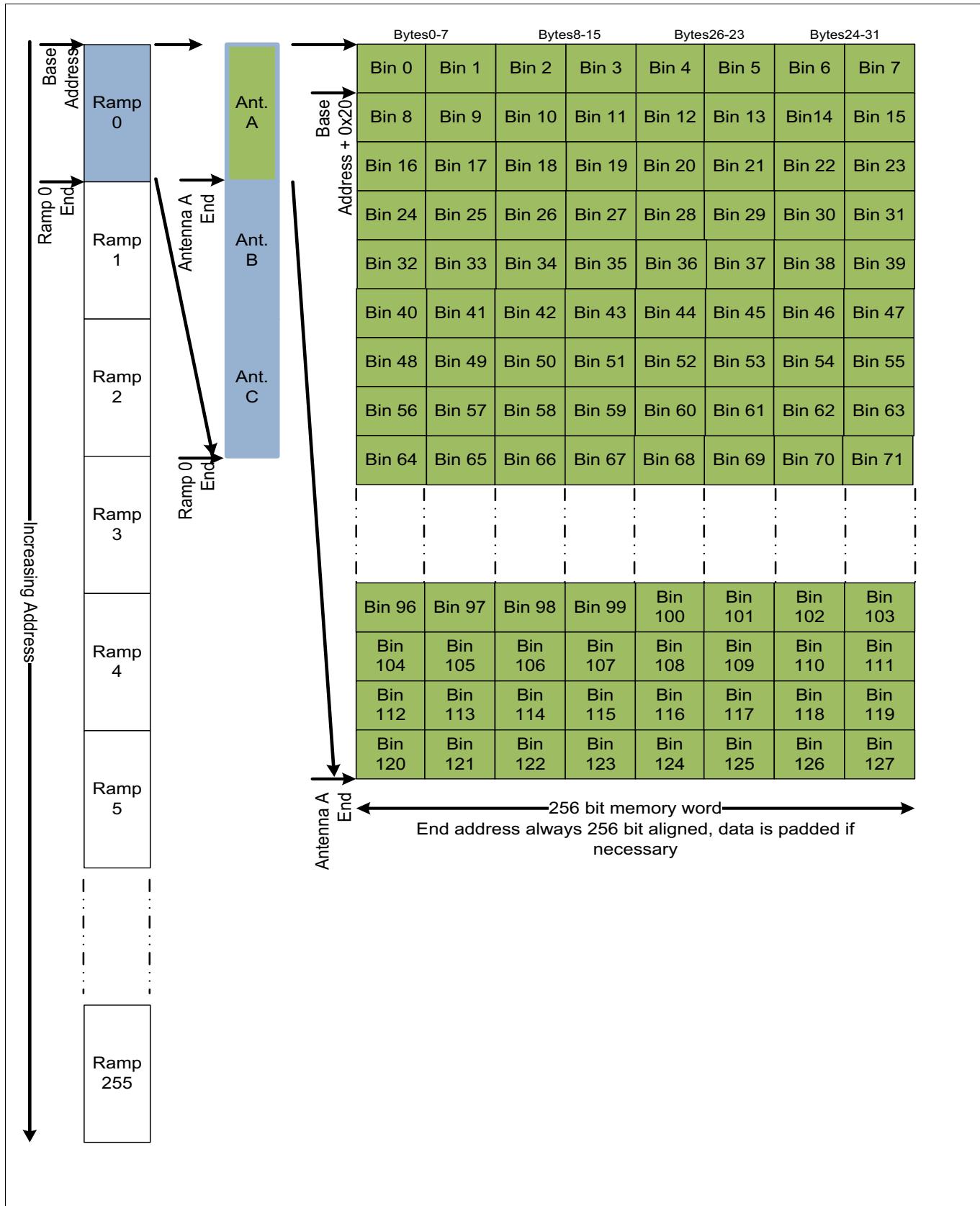


Figure 239 First Stage FFT Memory Organisation, 16 bit data, 3 antennae

## Signal Processing Unit (SPU)

### 19.2.8.14 Data Read Sequence for 2nd Stage FFT with 3 antennae and 16 bit operands

The figure below is showing Input DMA Unit read sequence for transpose before 2nd stage FFTs with 16bit operands and 3 antenna

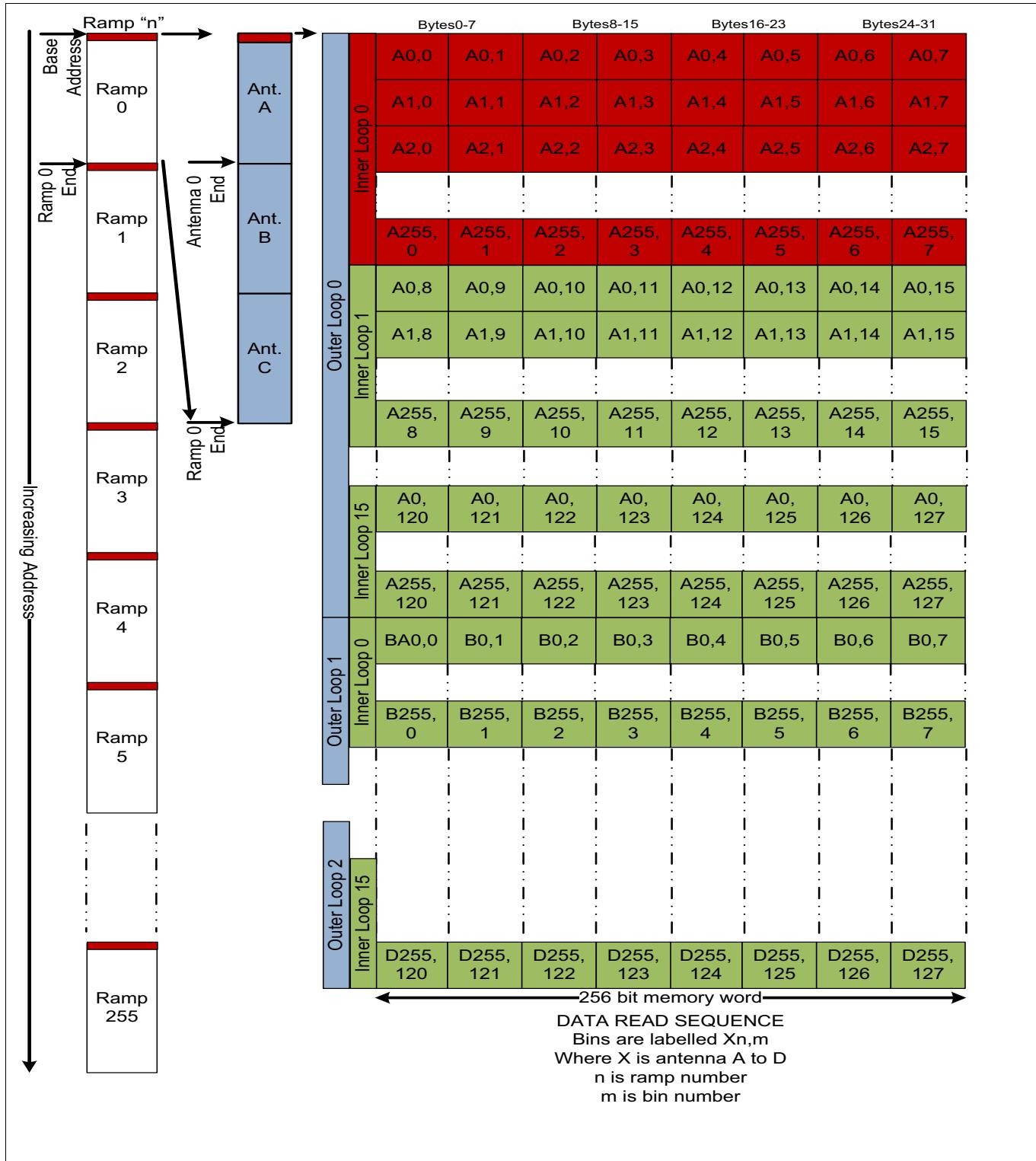


Figure 240 Transpose read order for 2nd stage FFT, 16 bit data, 3 antennae

this shows we can read 8 consecutive bins from the same antenna, do the transpose and process all of them.

## Signal Processing Unit (SPU)

### 19.2.8.15 Data Mapping for 2nd Stage FFT results with 16bit Operands and 3 Antennae

The figure below is showing Radar memory for 16bit operands and 3 antennae and shows 256 bit word content after 2nd stage FFT. This assumes no in place FFT with transposed addressing.

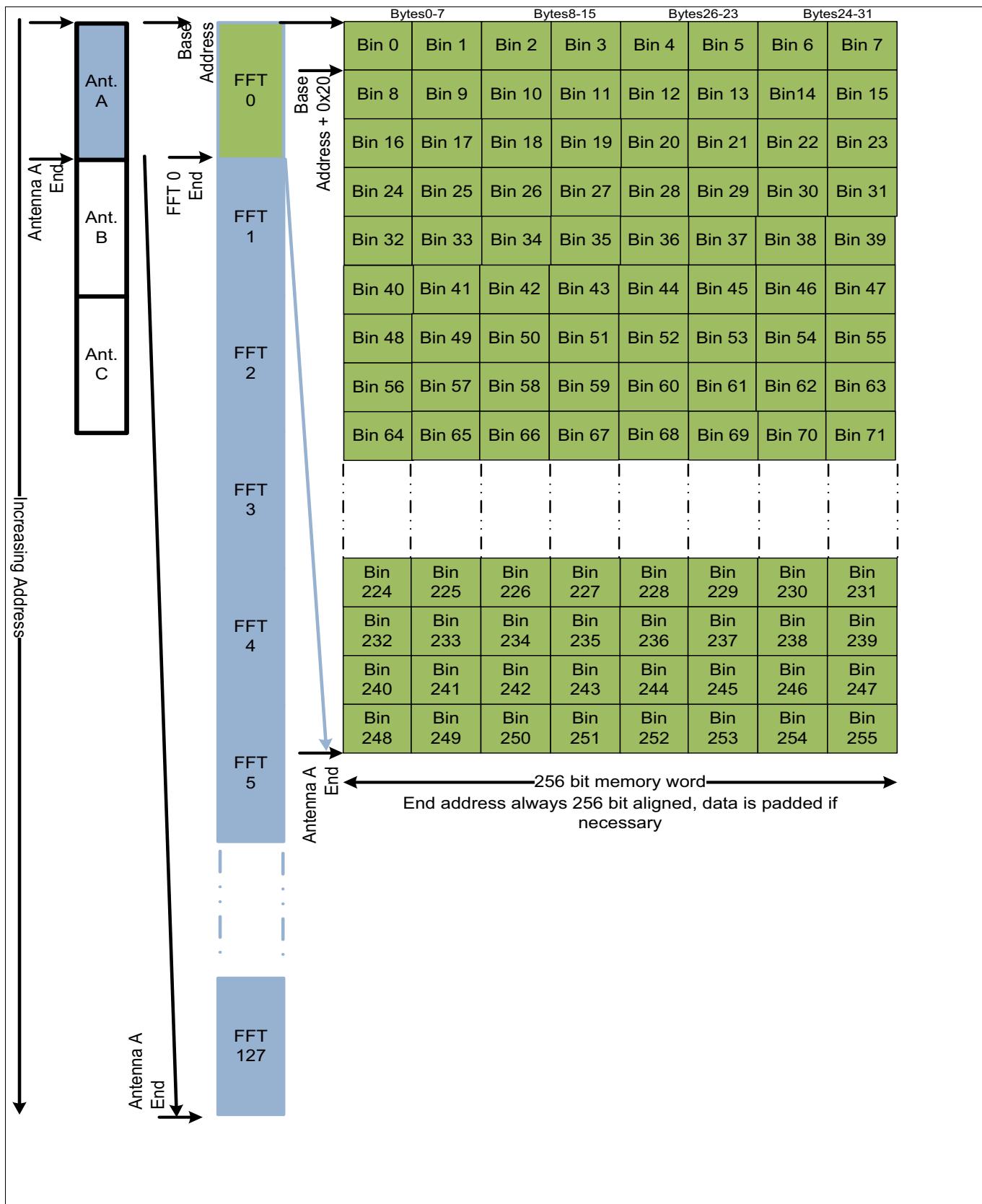


Figure 241 2nd Stage FFT Data Organisation, 3 antennae with 16 bit operands

## Signal Processing Unit (SPU)

### 19.2.8.16 Data Read Order in Integration Mode

The following figure shows the Input DMA Unit read sequence for 16 bit data and 3 antennae when coherent or non-coherent integration across the antennae is required. This shows individual reads and is not optimised for bandwidth. See [Chapter 19.2.8.4, Bandwidth Optimised Integration Mode](#) for details on bandwidth optimisation.

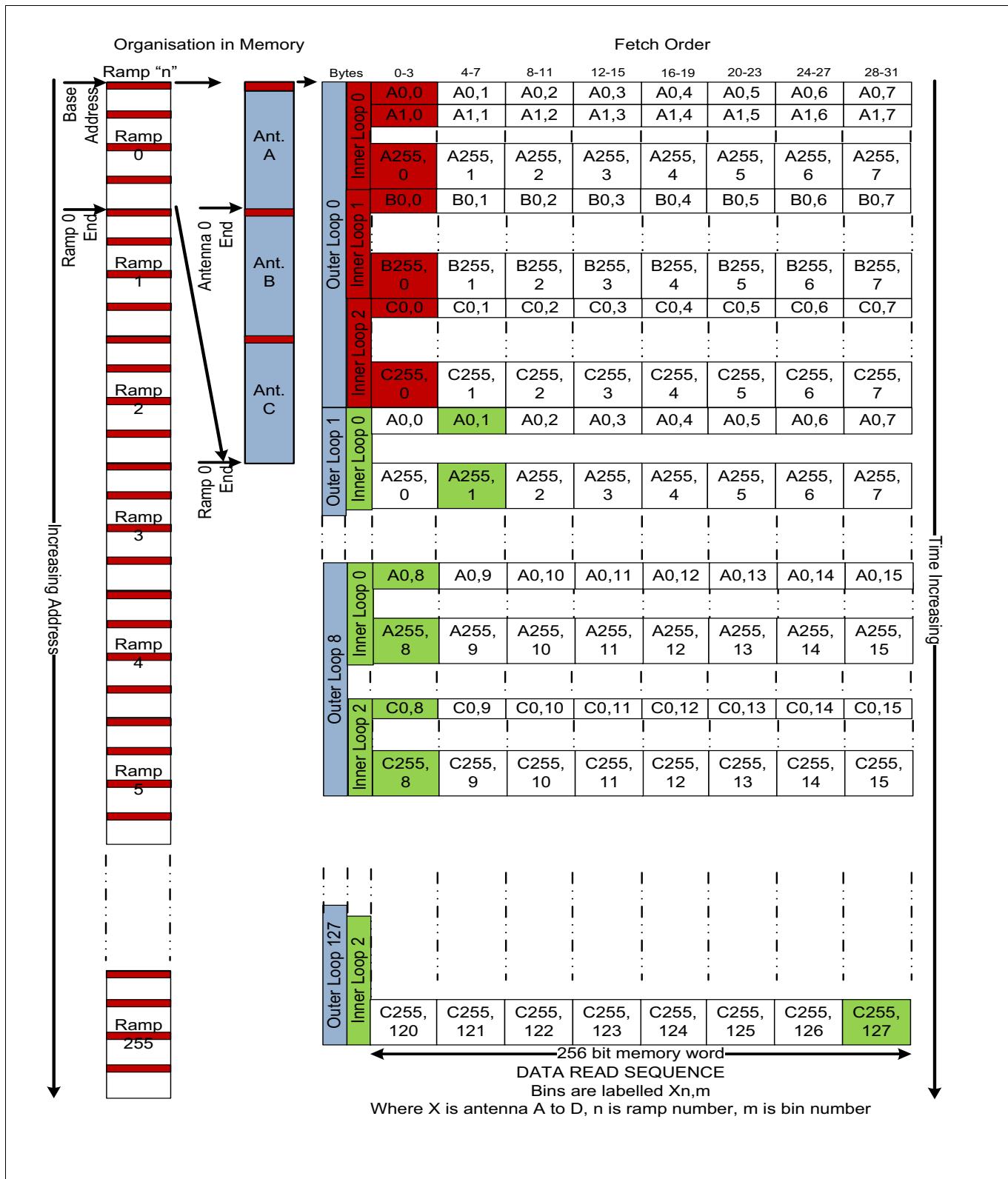


Figure 242 Integration Mode Transpose read order for 2nd stage FFT, 16 bit data, 3 antennae

## Signal Processing Unit (SPU)

### 19.2.8.17 Data Mapping for 2nd Stage FFT results with 16bit Operands and 3 Antennae

The figure below is showing Radar memory for 16 bit operands and 3 antennae and shows 256 bit word content after 2nd stage FFT. This assumes no “in place FFT” with transposed addressing.

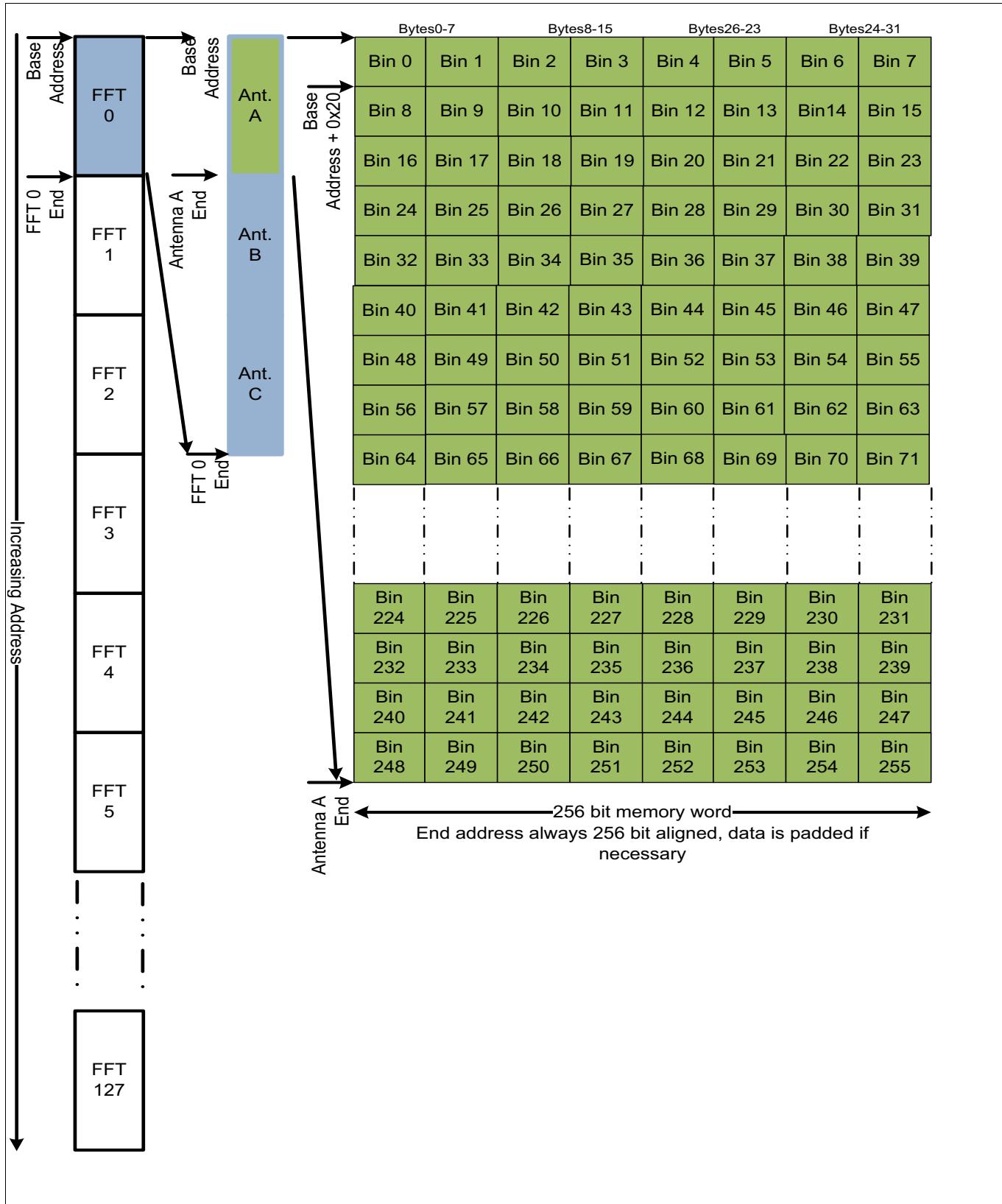
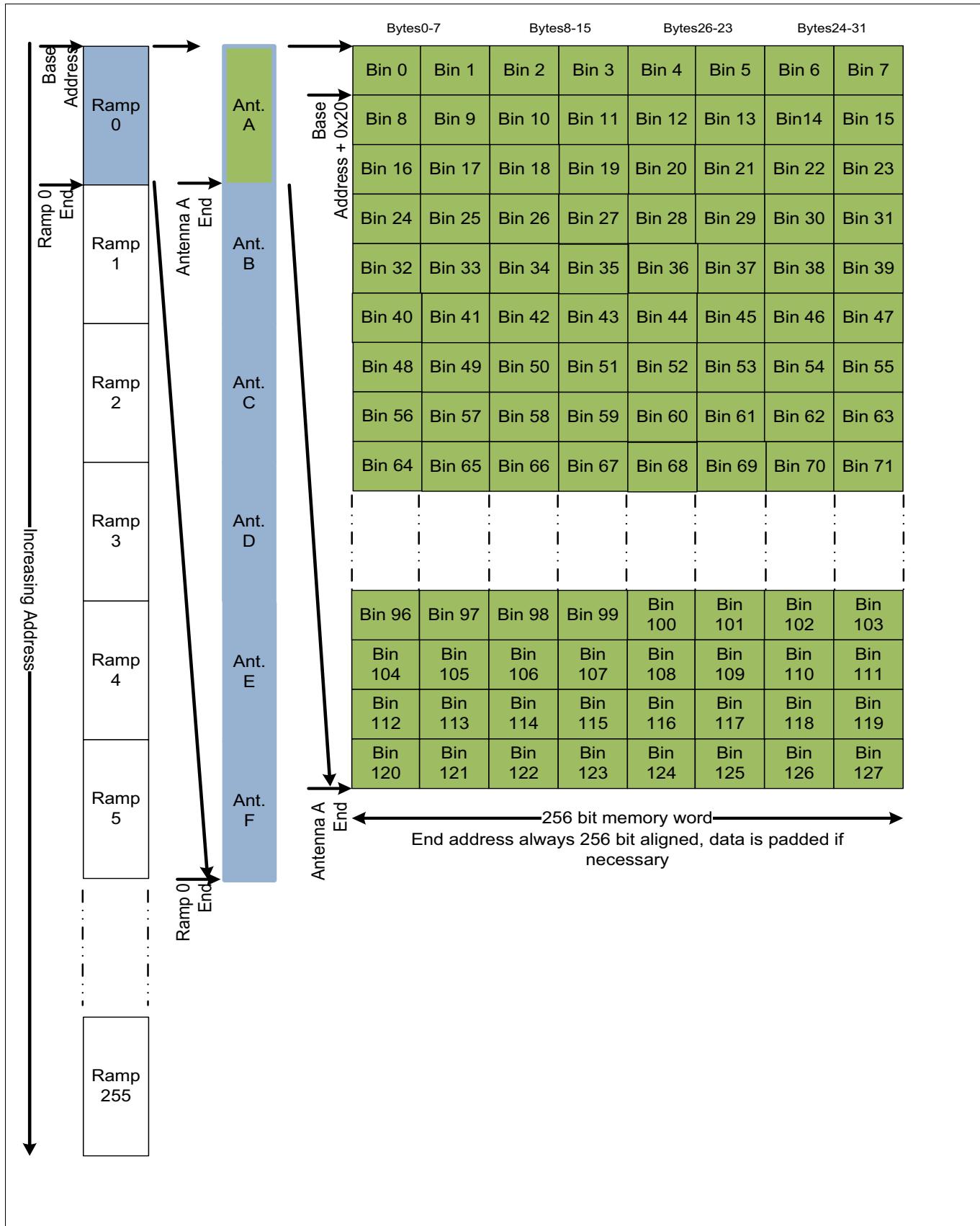


Figure 243 Integration Mode, 2nd Stage FFT Data Organisation, 3 antennae with 16 bit operands

## Signal Processing Unit (SPU)

### 19.2.8.18 Memory mapping for 6 antenna and 16bit operands

The figure below is showing the content of the 256bit words in Radar memory after 1st stage FFT.



**Figure 244 First Stage FFT Memory Organisation, 16 bit data, 6antennae**

## Signal Processing Unit (SPU)

### 19.2.8.19 Data Read Order for 2nd Stage FFT with 6 antennae and 16 bit data

The figure below is showing Input DMA Unit read sequence for transposed addressing before 2nd stage FFTs with 16bit operands and 6 antennae

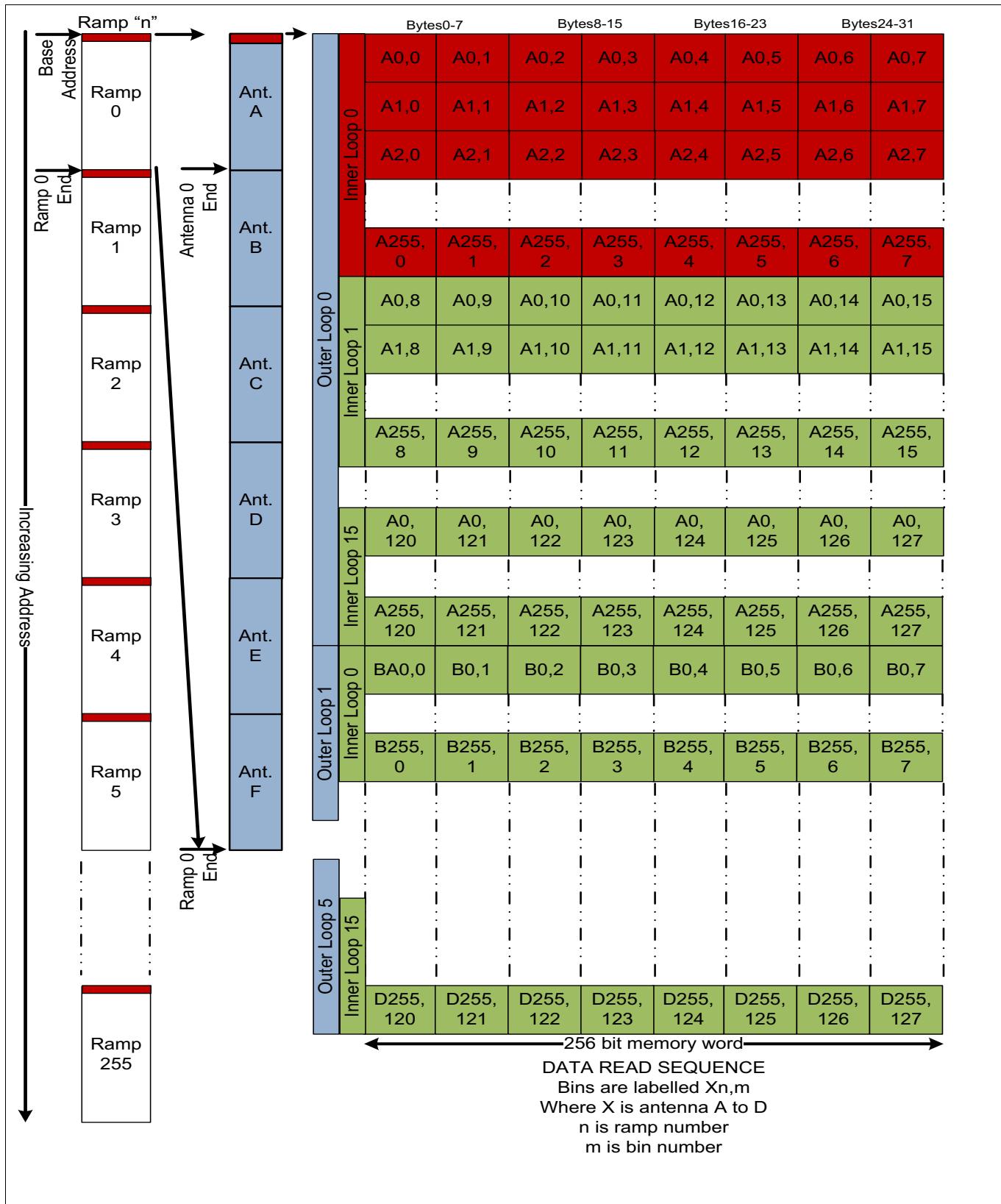


Figure 245 Transpose read order for 2nd stage FFT, 16 bit data, 6 antennae

## Signal Processing Unit (SPU)

### 19.2.8.20 Data Mapping for 2nd Stage FFT results with 16bit Operands and 6 Antennae

The memory map after 2nd stage DMA memory requirements to be optimised.

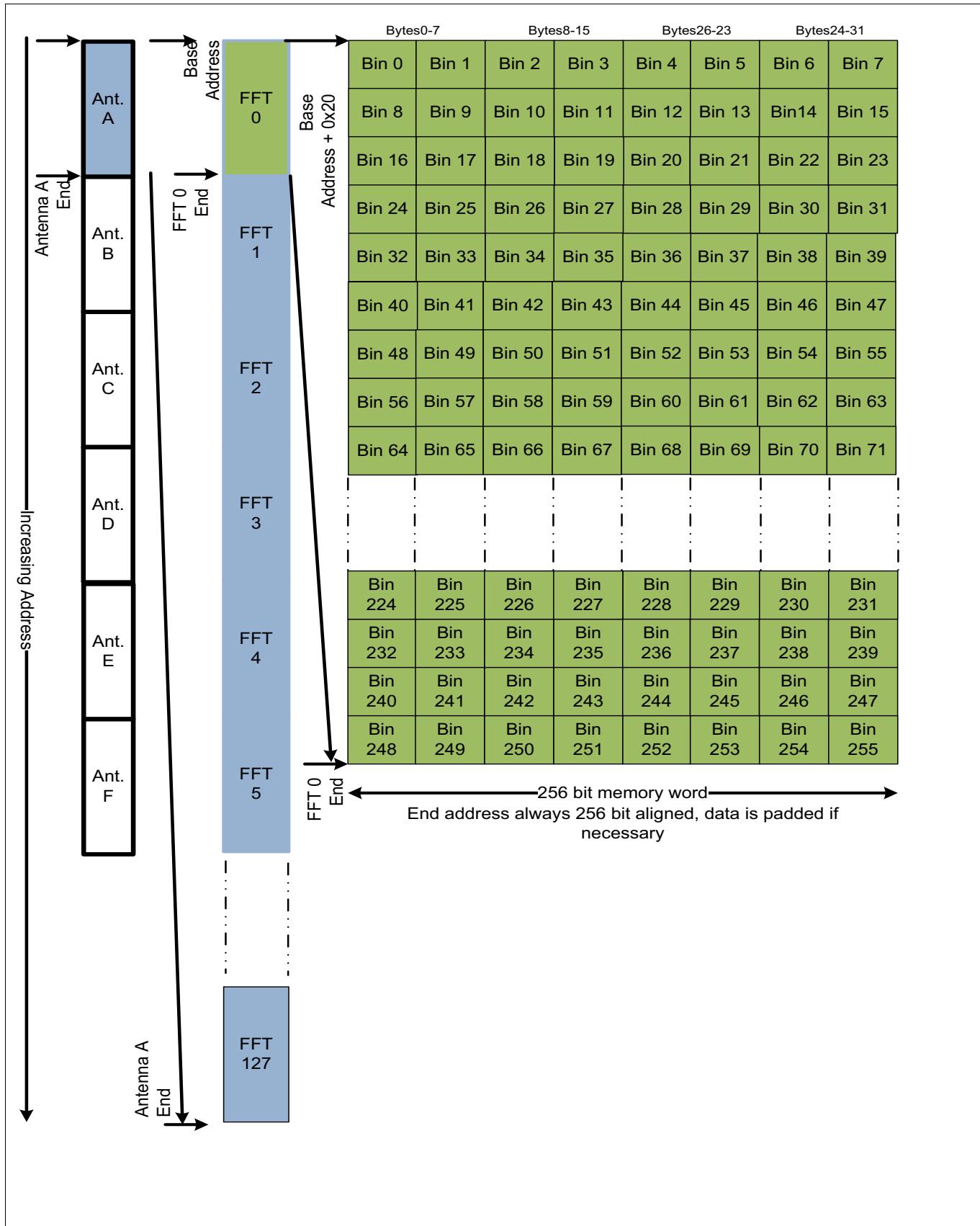
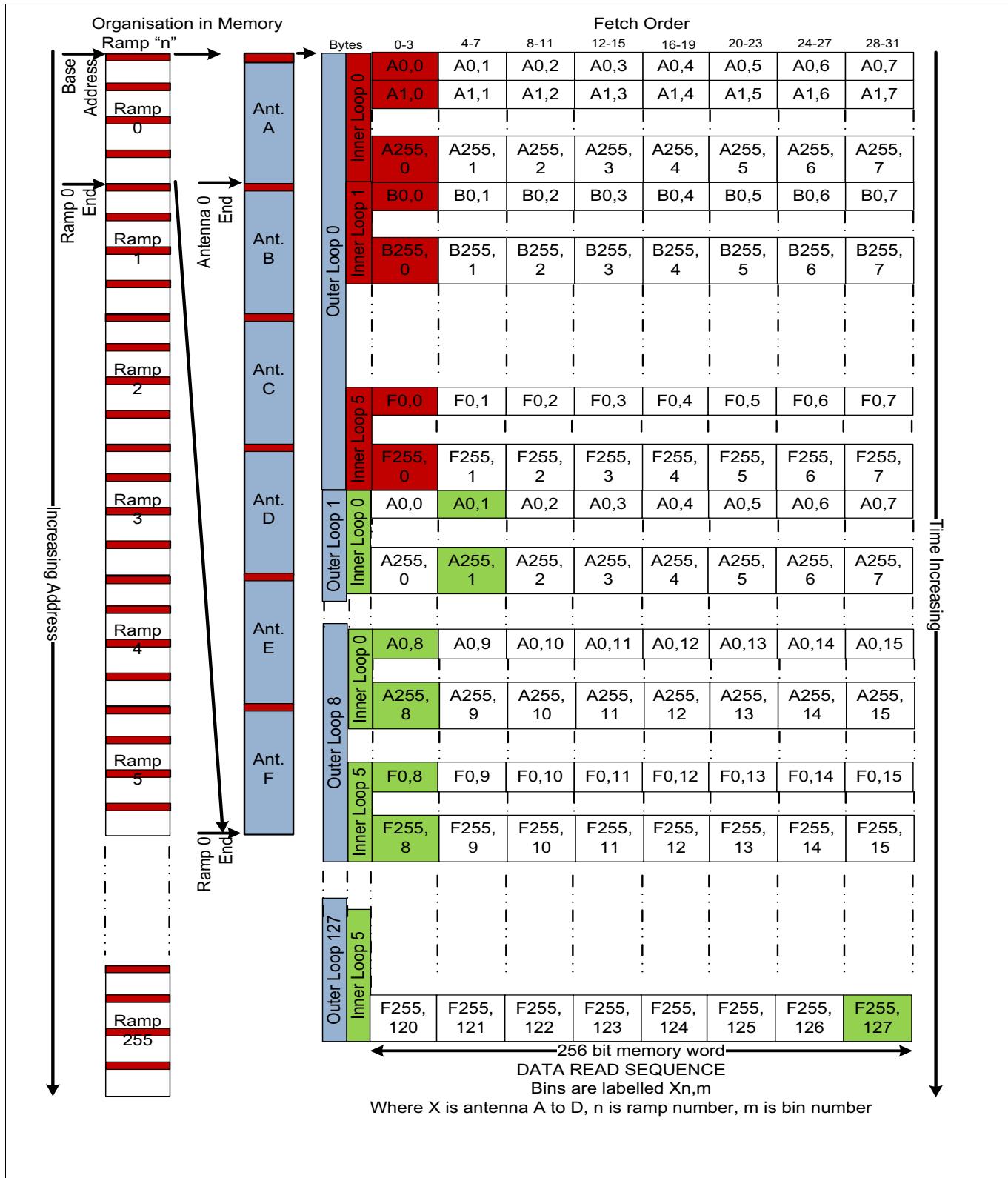


Figure 246 2nd Stage FFT Data Organisation, 6 antennae with 16 bit data

## Signal Processing Unit (SPU)

### 19.2.8.21 Data Read Sequence in Integration Mode

The following figure shows the Input DMA Unit read sequence for 16 bit data and 6 antennae when coherent or non-coherent integration across the antennae is required. This shows individual reads and is not optimised for bandwidth. See [Chapter 19.2.8.4, Bandwidth Optimised Integration Mode](#) for details on bandwidth optimisation.



## Signal Processing Unit (SPU)

### 19.2.8.22 Integration Mode Data Mapping for 2nd Stage FFT results with 16bit Operands and 6 Antennae

The figure below is showing Radar memory layout for 16 bit operands and 6 antennae and shows 256 bit word content after a 2nd stage FFT. This assumes no “in place FFT” with transposed addressing.

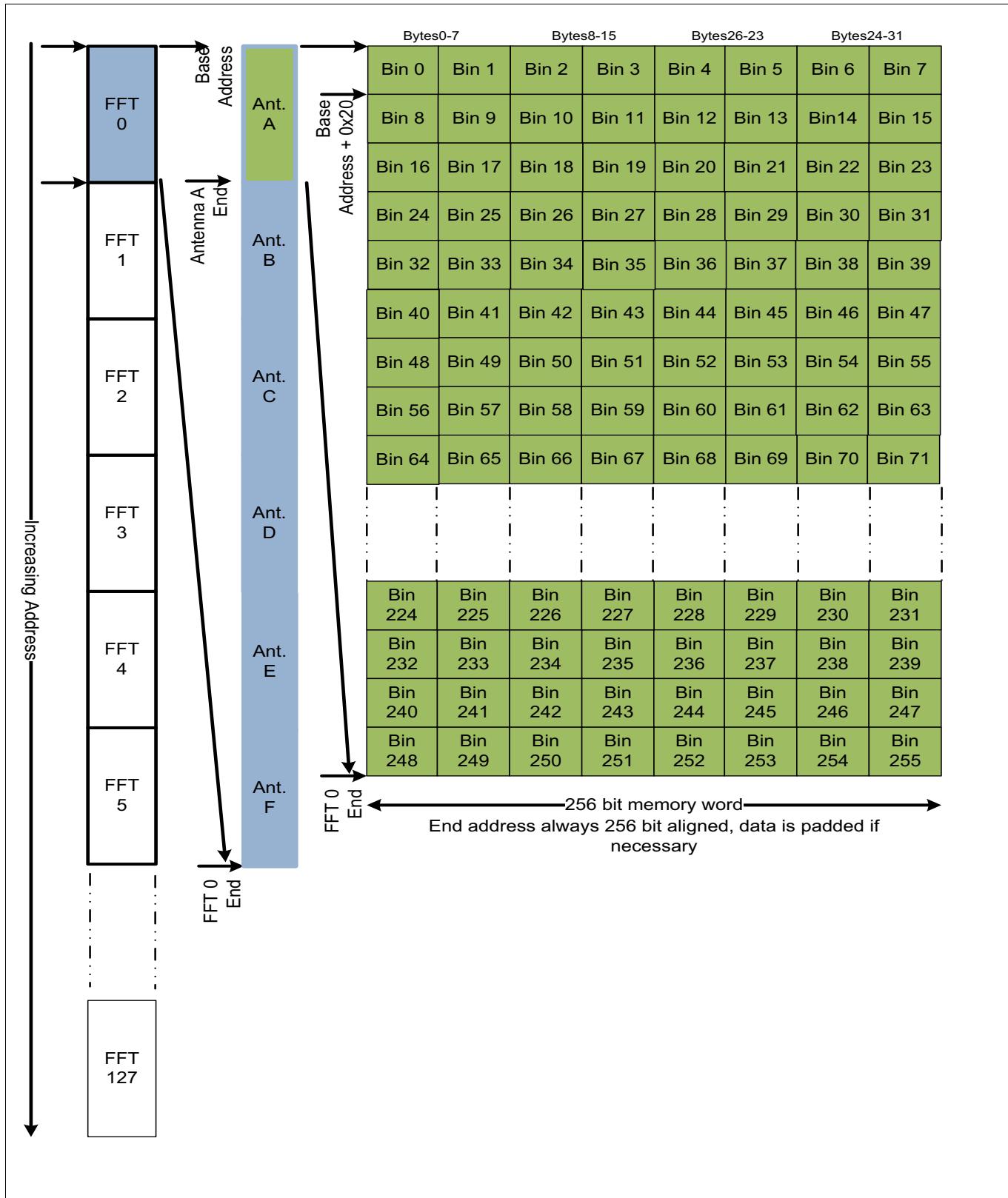


Figure 248 Integration Mode, 2nd Stage FFT Data Organisation, 6 antennae with 16 bit operands

## Signal Processing Unit (SPU)

### 19.3 Functional Description

This section describes programming and using the SPU. It deals with each of the major functional blocks of the SPU in the order data flows through the processing pipeline.

#### 19.3.1 Input DMA Engine

The Input DMA Engine can either passively accept data pushed by the ADC interfaces (RIF or Radar Interface) or actively load part or all of an existing data cube from the Radar Memory. Its function is to structure the data in the Buffer RAM into FFT data sets. To do this, it uses the following general control data as well as control information specific to the data source used.

**Table 596 Input DMA Unit General Parameters for all Input Sources**

Parameter	Definition	Comments
Data Source	ADC IF 0, ADC IF 1, both ADC IF or Radar Memory	<b>ID_CONF.SRC</b>
start mode	trigger mode for the SPU ( <b>CTRL.MODE</b> ). Options are defined in <a href="#">Table 627, CTRL.MODE Value Definitions</a> :	

Please refer to [Section 19.2.8, Memory mapping](#) for background information.

#### 19.3.1.1 Load ADC data from the RIF

When loading data from the RIF or RIFs, processing mode is always default and the input data format and number of antennae must be aligned with the RIF configuration settings. In addition the following operating modes can be configured

**Table 597 Input DMA Unit Specific Parameters for ADC Data**

Parameter	Definition	Comments
Ramps per Measurement Cycle	Number of ramps to be acquired after trigger.	Excess ramps will be ignored unless the SPU is retriggered <b>ID_CONF.RAMPS</b>
Number of active antenna	Number of physical antennae connected to the ADC interfaces of the microcontroller	<b>ID_CONF.ANT</b>
Input format	Signed or unsigned	<b>ID_CONF.SIGNED</b>
Samples per Ramp or Bins per FFT	The number of sample points to be used in the input datasets for the FFT	<b>ID_CONF.SMPLCNT</b>
Input format	Real or complex,	<b>ID_CONF FORMAT</b>

#### 19.3.1.1.1 Principles of Operation (ADC IF)

There are some restrictions imposed on the physical configuration of the overall system by the Input DMA. The most significant of these are that, if both ADC IF<sup>1)</sup> instances are being processed by a single SPU:

- the configuration of the ADC IFs must be identical so that the quantity of data being fed to the SPU by each ADC IF is the same.

1) The ADC IF function is implemented in the RADAR Interface (RIF) module

## Signal Processing Unit (SPU)

- The data sources providing data to the ADC IFs must be synchronised. This is for two reasons
  - To enable meaningful integration (or other calculations) across the antennae, the sampling time for each antenna must be aligned and kept consistent to within very small limits
  - The ADC IFs contain buffers to ensure that the data fed to the SPU is aligned in the same clock cycle for equivalent antenna. The alignment of data is triggered by setting a configuration option in the RIF module registers. The data provided to the ADC IFs must be sufficiently aligned to prevent these buffers overflowing. The buffering is sufficient to allow for one complete sample misalignment at the serial inputs to the ADC IFs.

When configured for internal or software trigger mode, the SPU will accept all data from the ADC IFs. The ADC IF in this case must maintain synchronisation to the ramps and only pass valid data to the SPU (although there is a limited facility to remove undesired data by truncating the dataset in the MATH1 unit).

When configured for external trigger, the SPU will accept trigger pulses from the SPU Lockstep module and use these to maintain synchronisation. The trigger output from the SPU Lockstep module is set by writing to a field in a register and is connected to both SPUs. This allows both SPUs to be triggered simultaneously.

### Antennae (ID\_CONF.ANT)

The number of active antennae is the number of physical antennae attached to the configured data source (ADC IF0, ADC IF1 or both). For real data this can be up to four antennae per ADC IF. However for complex data, each antenna will require two data channels, one for the real data component and one for the complex data component. This halves the number of physical antennae that can be connected to each ADC IF to a maximum of two.

#### 19.3.1.1.2 Split Processing

This mode is used in systems with 2 RIF instances when there is too much data from the ADCs to be handled by the lockstepped SPUs in a single pass. ID\_CONF.SRC should be set to  $10_B$  (BOTH) and the antennae must be distributed symmetrically between the two RIF instances.

The data from RIF0 is processed and the data from RIF1 is written directly to Radar Memory for later processing. The address is stored in the ID\_CONF2.BPADDR field. The first pass is configured by setting ID\_CONF2.BYPASS with ID\_CONF2.BPRLD (Bypass Reload) cleared. This will process the RIF0 data and bypass the RIF1 data. The second pass is configured by setting both ID\_CONF2.BYPASS and ID\_CONF2.BPRLD. This will read the data from the address in the BPADDR field rather than the RIF interface.

All other register settings, apart from the ODP\_CONF.BASE should be the same. In particular, IDM\_CONF.SRC should remain at  $10_B$  to ensure that the IDM uses the correct format when loading the data. In Bypass Mode, the reloaded data is treated as an ADC IF (RIF) data stream rather than a datacube in Radar Memory. The same parameters therefore apply as when loading ADC data directly from a RIF. The parameters used when loading from Radar Memory are not applicable.

#### 19.3.1.2 Load from Radar Memory

The following options are specific to loading data from Radar Memory:

## Signal Processing Unit (SPU)

**Table 598 Input DMA Unit Parameters for Reading Data from Radar Memory**

Parameter	Definition	Comments
Input format	16bit or 32bit precision complex data, 16bit or 32bit precision real data, IEEE 754 half precision floating point real data, IEEE 754 half precision floating point complex data, or 32 bit power	<b>ID_RM_CONF FORMAT</b>
DMA base address	DMA base address when starting a new FFT sequence from Radar memory	Word (256 bit) address relative to start of Radar Memory <b>ID_RM_CONF BASE</b>
DMA outer loop address offset	Offset address to be added to base address (Radar Memory address calculation)	Byte offset <b>ID_RM_OLO.OLO</b>
DMA outer loop repeat value (parameter “m”)	Number of times of outer loop execution	e.g. for 2nd stage FFT, this would be the number of antennae to process in default mode. <b>ID_RM_IOLR.OLR</b>
DMA inner loop address offset	Offset address to be added to base address (Radar Memory address calculation)	Byte offset <b>ID_RM_ILO.ILO</b>
DMA inner loop repeat value (parameter “n”)	Number of times of inner loop execution	Inner loop count is incremented by one every time a complete FFT is read from memory <b>IDM_RM_IOLR.ILR</b>
Bin offset	Radar Memory address offset between adjacent bins in an input dataset	Byte offset. When reading a linear FFT from memory, this would be set to the size of one data element <b>IDM_RM_BLO.BLO</b>
Bin Offset Loop Repeat (parameter “p”)	Number of reads needed to traverse one input FFT. (number of bins in an input dataset)	The number of samples in each FFT input (dataset) being constructed <b>IDM_RM_BLR.BLR</b>
Addressing Mode	Linear or Transpose Addressing	<b>ID_RM_CONF TRNSPS</b>
Processing Mode	Default Mode or Integration Mode for bandwidth optimisation.	Integration Mode should be used when construct input datasets for use with the coherent or non-coherent integration functions in the MATH2 units <b>IDM_RM_CONF PM</b>

## Signal Processing Unit (SPU)

**Table 598 Input DMA Unit Parameters for Reading Data from Radar Memory (cont'd)**

Parameter	Definition	Comments
Number of simultaneous data blocks from RAM	For integration mode, defines the number of concurrent information sets (data blocks) that are fetched from memory.depends on memory organization.Number from 1 to 8.(1,2 4 and 8 supported).	Number of data blocks that will fit into the buffer memory. <b>IDM_RM_CONF.BLOCKS</b> Add one to this field to get the number of data blocks. Supported values are therefore: 0, 1, 3, and 7.
Mapping of FFT datasets to antenna	Defines how each dataset in the buffer memory is linked to an antenna ID for windowing and other antenna specific operations	<b>ID_RM_CONF.AM</b>

All address fields in registers are specified in words (256 bits or 32 bytes).

At any time the byte address that the Input DMA is reading from is defined as:

```
(base address<<5) + ((outer loop offset)*m) + ((inner loop offset)*n) + ((bin loop offset)*p)
```

where **n** and **m** are as defined above and **p** is the bin being read. This will be used to generate a word address for Radar Memory by truncating bits [4:0] of the generated address. Bits [4:0] will be used to select within the 256 bit word retrieved from radar Memory.

The input DMA engine will read (at least) one complete set of FFT input data per inner loop using the Bin Offset field to increment the byte address between sample reads.

See [Chapter 19.2.8.3.1](#) and [Chapter 19.2.8.4](#) for an overview of the effect of the Processing Mode options on memory usage. A more detailed description of the operating modes follows

### 19.3.1.2.1 Principles of Operation (Radar Memory)

The Input DMA is designed to allow autonomous processing of a three dimensional data array stored in the Radar memory. Typically the three axes of the array would be representing FFT sample number, ramp or acquisition count and antenna. The intention is to allow assembling input sets for the FFT accelerator by stepping across the axes in any arbitrary order.

This is accomplished by constructing the read address for the Input DMA from a base address and three independent address offsets (“bin loop address”, “inner loop address” and “outer loop address”).

The term “bin loop” is used as the bin loop repeat value maps directly to the number of bins in each FFT dataset constructed in the buffer RAM. The bin loop repeat value defines the size of the input dataset.

**Note:** *The maximum number of bins supported in an FFT dataset is 2048. Setting a Bin Loop Repeat value for an FFT size greater than this (i.e. ID\_RM\_BLR.BLR > 2047) is not supported and should be avoided.*

Some assumptions are made about the organisation of the data. Specifically that:

- Each dataset in memory starts at a 32 byte aligned address. This allows the bandwidth optimisation hardware to function and is enforced by the Output Data Manager. This restriction has to be observed when using software to build data arrays in memory for reading by the Input Data Manager
- The second assumption follows from the first. If all datasets are 32 byte aligned, it follows that 2 co-ordinate axes of the data array must be multiples of 32 byte. Only a single co-ordinate axis, the one used to step along data samples can be less than a multiple of 32 bytes and this must be set to the data sample size. The other axes will be used to step along datasets.

## Signal Processing Unit (SPU)

**Note:** The SPU is specified to cope with Radar Memory sizes up to 16 MiB. If programming causes it to generate an address outside the range of the available Radar Memory in the product, an error will be generated by the Input Data Manager

Each counter has an associated offset and a repeat count. Operation is then as follows:

**Table 599 Input DMA Execution Flow**

1	initialise the “bin loop address”, the “inner loop address” and the “outer loop address”.
2	loop
3	The read address is computed by adding the “base address”, the “bin loop address”, the “inner loop address” and the “outer loop address”.
4	An input value is then read from the address
5	“bin loop count” is incremented
6	If { the bin loop count has reached the “bin offset loop repeat” value } then
7	“bin loop count” is reset
8	“bin loop address” is reset
9	“inner loop count” is incremented
10	If { the “inner loop count” has reached the “inner offset loop repeat” value } then
11	“inner loop count” is reset
12	“inner loop address” is reset
13	“outer loop count is incremented”
14	If { the “outer loop count” has reached the “outer offset loop repeat” value } then
15	execution has completed, exit loop
16	else
17	“outer loop address” is set to “outer loop address” plus “outer loop offset”
18	end if
19	else
20	“inner loop address” is set to “inner loop address” plus “inner loop offset”
21	end if
22	else
23	“bin loop address” is set to “bin loop address” plus “bin loop offset”
24	end if
25	end loop
26	stop

The flow above is the basic operation flow without considering bandwidth optimisation. As the Radar Memory is organised as 256 bit words and the data element occupy either 32 or 64 bits depending on precision and format, then bandwidth usage can be optimised by using as much as possible of the 256 bit word<sup>1)</sup>. The Input DMA will attempt to do this based on register settings by “collapsing” one or more of the loops i.e. effectively performing multiple iterations of one of the loops in a single pass as described in the following sections.

1) The exceptions to this are the 16 bit precision, real only formats, REAL16BIT and REAL16FP. These formats can only be used if TRNSPS=0. If the SPU is configured with (FORMAT=REAL16BIT or FORMAT=REAL16FP) and TRNSPS=1, the resulting datasets will be corrupted.

## Signal Processing Unit (SPU)

**Table 600 Input DMA Loop Mapping**

Mode	Co-ordinate Mapping <sup>1)</sup>			Notes	ID_RM_CONF Register Settings <sup>2)</sup>
	Bin Loop Offset	Inner Loop Offset	Outer Loop Offset		
1	<b>Sample</b>	FFT	Antenna	Linear (Default) Addressing Mode. Each Radar Memory Read fetches multiple samples for the same FFT. Integration mode can be used to build a datablock for NCI or DBF <sup>3)</sup> . Blocks must be $0_B$ See <a href="#">Chapter 19.3.1.2.2</a>	<b>TRNSPS=0</b> PM=0,1 <b>BLOCKS=0</b>
2	<b>Sample</b>	Antenna	FFT	Linear (Default) Addressing Mode. Each Radar Memory Read fetches multiple samples for the same FFT. Integration mode can be used to build a datablock for NCI or DBF. Blocks must be $0_B$ See <a href="#">Chapter 19.3.1.2.2</a>	<b>TRNSPS=0</b> PM=0,1 <b>BLOCKS=0</b>
3	FFT	<b>Sample</b>	Antenna	Transpose Addressing Mode. Each Radar Memory Read fetches samples for multiple FFT datasets. Integration mode is not supported. Blocks must be $0_D$ See <a href="#">Chapter 19.3.1.2.3</a> . ID_RM_CONF.FORMAT=REAL16BIT not supported. ID_RM_CONF.FORMAT=REAL16FP not supported.	<b>TRNSPS=1</b> PM=0 <b>BLOCKS=0</b>
4	FFT	Antenna	<b>Sample</b>	Transpose Addressing Mode. Each Radar Memory Read fetches samples for multiple data blocks. Integration mode must be set. Blocks can be set to a value other than $0_D$ for bandwidth optimisation See <a href="#">Chapter 19.3.1.2.4</a> ID_RM_CONF.FORMAT=REAL16BIT not supported. ID_RM_CONF.FORMAT=REAL16FP not supported.	<b>TRNSPS=1</b> PM=1 BLOCKS=0,1,3,7
5	Antenna	<b>Sample</b>	FFT	Transpose Addressing Mode. Each Radar Memory Read fetches samples for multiple FFT datasets. Integration mode is not supported. Blocks must be $0_D$ . See <a href="#">Chapter 19.3.1.2.3</a> ID_RM_CONF.FORMAT=REAL16BIT not supported. ID_RM_CONF.FORMAT=REAL16FP not supported.	<b>TRNSPS=1</b> PM=0 <b>BLOCKS=0</b>
6	Antenna	FFT	<b>Sample</b>	Transpose Addressing Mode. Each Radar Memory Read fetches samples for multiple data blocks. Integration mode must be set. Blocks can be set to a value other than $0_D$ for bandwidth optimisation. See <a href="#">Chapter 19.3.1.2.4</a> ID_RM_CONF.FORMAT=REAL16BIT not supported. ID_RM_CONF.FORMAT=REAL16FP not supported.	<b>TRNSPS=1</b> PM=1 BLOCKS=0,1,3,7

- 1) Incrementing the Offset counter increments address to next sample bin, next FFT dataset or next antenna
- 2) Mandatory settings are shown in bold. Options where multiple values are possible are shown as a commas separated list
- 3) Provided that ID\_RM\_IOLR.ILR is less than or equal to  $7_D$ .

## Signal Processing Unit (SPU)

### 19.3.1.2.2 Case 1: “Bin Offset” is set to Sample Size

In this case, the bins in Radar Memory to be used as consecutive inputs bins for each FFT dataset are stored in consecutive addresses in Radar Memory so each 256 bit read will load four, eight or sixteen bins depending on the precision of the data. The input DMA will therefore execute four, eight or sixteen iterations of the “bin offset” loop simultaneously. This is the [“Default Memory Mode” on Page 11, Section 19.2.8.3.1](#). As this does not require address transposition, the ID\_RM\_CONF.TRNSPS bit must be set to  $0_B$  (LIN).

Bandwidth optimisation is not needed in this mode as all data fetched in a 256 bit read is used in the same FFT dataset.

#### Buffer Memory Switching

If the ID\_RM\_CONF.PM bit is set to IM, this signals that the data block should contain one dataset for each antenna (i.e. the Inner Loop Offset is set to step across antennae) and the Buffer Memory will be switched when the Inner Loop Repeat Count is reached. Otherwise, the Input Buffer Memory will be switched every four (for 64 bit data) or eight (for 32 bit data) iterations of the Inner Loop or when all data has been read whichever occurs first.

**Note:** *Setting ID\_RM\_CONF.BLOCKS to a non-zero value with ID\_RM\_CONF.TRNSPS set to  $0_B$  is not supported and will result in undefined operation of the input DMA.*

### 19.3.1.2.3 Case 2: “Inner Loop Offset” is set to Sample Size

This is the first of the co-ordinate transposition cases, so the ID\_RM\_CONF.TRNSPS bit must also be set to explicitly signal the intent to use co-ordinate transposition to the logic of the SPU. The Bin Loop offset can be set to step either across antennae or across FFT result datasets.

The ID\_RM\_CONF.TRNSPS bit signals that the data from each 256 bit read will be used in different FFT input datasets and should therefore be written to different areas of the buffer memory (“scattered”).

As stated, the bins in Radar Memory to be used as consecutive input bins in each FFT dataset are stored in non-consecutive addresses in Radar Memory so each 256 bit read will load four or eight bins depending on the precision of the data<sup>1)</sup> but these samples will be from different FFT input sets. The input DMA will therefore execute four or eight iterations of the “inner loop” simultaneously. This bandwidth optimisation is described in [Section 19.2.8.3.1, “Default Memory Mode” on Page 11](#) and [Section 19.3.1.2.5, “Bandwidth Optimisation for Default Processing Mode” on Page 43](#).

ID\_RM\_CONF.PM must be set to DM for this operating case as the data from each 256 bit read is all needed for a single datablock.

#### Buffer Memory Switching

The Input Buffer Memory will be switched either when the Inner Loop Count is updated. i.e. every four (for 64 bit data) or eight (for 32 bit data) iterations of the Inner Loop or when all data has been read.

### 19.3.1.2.4 Case 3: “Outer Loop Offset” is set to Sample Size

This is the second of the co-ordinate transposition cases, so the ID\_RM\_CONF.TRNSPS bit must also be set to explicitly signal the intent to use co-ordinate transposition to the logic to the logic of the SPU. Either the Bin Loop Offset or Inner Loop Offset will be set to step across the FFT datasets stored in the Radar Memory. However in both of these cases, consecutive samples to be used as inputs to the FFT are stored in non-consecutive addresses in Radar Memory so, while each 256 bit read will load four or eight bins<sup>1)</sup> depending on the precision of the data, these samples will be from different FFT input sets. This differs from case 2 in that using the remaining data from

<sup>1)</sup> 16 bit precision real data (FORMAT=REAL16BIT) and 1 bit half precision floating point real data (FORMAT=REAL16FP) are not supported with TRNSPS=1.

## Signal Processing Unit (SPU)

the 256 bit read requires an increment of the outer loop. This is signalled to the logic by setting ID\_RM\_CONF.PM to IM ( $1_B$ ).

In this case, to avoid discarding data and wasting bandwidth, multiple data blocks can be built in memory by setting the ID\_RM\_CONF.BLOCKS bitfield to a value other than  $0_D$ . This bandwidth optimisation is described in **Section 19.2.8.4, “Bandwidth Optimised Integration Mode” on Page 12.** and **Section 19.3.1.2.6, “Bandwidth Optimisation Integration Processing Mode.” on Page 43.** This has the effect of collapsing the outer loop as well as the inner loop resulting in a three-dimensional array of input data in the buffer memory.

### Buffer Memory Switching

If the ID\_RM\_CONF.PM bit is set to IM, this signals that the data block should contain one dataset for each antenna (i.e. the Inner Loop Offset is set to step across antennae) and the Buffer Memory will be switched when the Inner Loop Repeat Count is reached. Note that in this case, the FFT accelerator will run ID\_RM\_CONF.BLOCKS+1 processing operations for each switch of the buffer memory. Otherwise, the Input Buffer Memory will be switched every four (for 64 bit data) or eight (for 32 bit data) iterations of the Inner Loop or when the Outer Loop Count is updated or all data has been read whichever occurs first.

### 19.3.1.2.5 Bandwidth Optimisation for Default Processing Mode

The memory mapping used to store data after the SPU processes ADC samples means that consecutive addresses in memory contain adjacent samples from the same FFT and that the same FFT result point from different antennae are always separated in the address space. Default mode is designed to work with this memory organisation to optimally construct multiple input datasets from each set of 256 bit reads. Each of the samples in a single data read will be allocated to different input datasets in the buffer memory.

The mode is enabled by setting ID\_RM\_CONF.TRNSPS=TRN and ID\_RM\_CONF.PM=DM

To work in default mode either the **ID\_RM\_BLO.BO** or **ID\_RM\_ILO.ILO** bitfields must be set to the sample size.

**Note:** One of the three fields, **ID\_RM\_OLO.OLO**, **ID\_RM\_ILO.ILO** and **ID\_RM\_BLO.BO** must always be set to the sample size.

### 19.3.1.2.6 Bandwidth Optimisation Integration Processing Mode.

Integration Mode Bandwidth Optimisation is intended for the use case where the MATH2 unit is required to integrate results across antennae while the input data manager is using transposed addressing. In this mode, the Input DMA Engine must load data into the Input Buffer Memory so that each constructed data block contains one dataset from each antenna.

The mode is enabled by setting ID\_RM\_CONF.TRNSPS=TRN and ID\_RM\_CONF.PM=IM.

In this case, the data block being constructed in memory would use only one sample point from each 256 bit read, as the first dimension of traversing the data cube (Bin Loop Offset) is across FFT results and the second (Inner Loop Offset) is across antennae (i.e. the data blocks constructed are suitable for use by the coherent or non-coherent integration functions of the MATH2 unit). This compares to the default transpose addressing mode, where the first dimension is still across FFT results but the second is across sample points and each read fetches several contiguous sample points which can be used to construct datasets that can be processed as a single data block.

To mitigate the inefficient use of Radar Memory bandwidth, an optimisation submode of Integration Mode controlled by the ID\_RM\_CONF.BLOCKS bitfield is available.

Enabling this submode allows multiple data blocks to be constructed in the buffer memory. Then, instead of switching the buffer memory between every data block, the base address the Loader Module uses for reads is adjusted instead. When IDM\_RM\_CON.PM is set to 1 and a non-zero value is written to ID\_RM\_CONF.BLOCKS, the

## Signal Processing Unit (SPU)

Input DMA Engine will try and use some of the excess data read to construct the additional data blocks in the buffer memory.

To support this, the Buffer RAM will be logically sub-divided into multiple segments based on the most significant address bits, one for each data block. The number of data blocks being equal to the value of the ID\_RM\_CONF.BLOCKS bitfield plus one. BLOCKS can be set to values between  $0_D$ ,  $1_D$ ,  $3_D$ , and  $7_D$ .

The Data Loader Unit will then read each data block sequentially rather than switching the buffer memory and only switch the buffer memory when all the data blocks have been processed. All units downstream of the FFT still process each data block discretely.

Application software must ensure that the data will fit into the sub-divided buffer memory.

**Note:** *If the number of data blocks built in the buffer memory does not use all of the 256 bits read from Radar Memory, then the same 256 bits will be read on successive passes until all the data is consumed. e.g. if the data in radar memory is 64 bit and BLOCKS is set to  $1_D$  then the Input DMA will make two passes through the data in Radar Memory. Each pass will create two data blocks in the Buffer Memory.*

### 19.3.1.3 Partial-acquisition Counter

The partial-acquisition counter is used to count ramps and hence to determine the absolute position in the measurement cycle. It is reset by software and will then use the configuration information (samples per ramp and number of antennae) valid at the point of reset to count ramps until the next time it is reset by software.

The intended use case of the counter is to allow the SPU to be triggered multiple times during a single measurement cycle while still retaining synchronisation with the input data stream from the Radar Interface(s). This allows the first few ramps of a measurement cycle to be processed to determine useful information about the input data. The SPU can then be reprogrammed to make most effective use of the input data and retriggered at a known ramp count to process the remaining data from the measurement cycle.

The counter can be configured to generate an error or attention interrupt if any particular value is reached with the counter still enabled.

The SPU can be configured to delay start of processing until a particular value is reached.

The operation of functions related to this counter are controlled by the bitfields of the PACTR register.

**Table 601 Partial-acquisition Counter Specific Parameters for ADC Data**

Parameter	Definition	Comments
Initialise the Counter	Reset the value of the partial-acquisition counter to $0_D$ .	<b>PACTR.RST</b>
Enable the Counter	Counter will increment every time the IDM requests a switch of the input buffers.	<b>PACTR.EN</b>
Limit Value for the counter	Comparison value for triggering a pre-programmed event Note that for comparison purposes, the first ramp of a measurement cycle is considered to be ramp 0.	<b>PACTR.LIMIT</b>
Trigger	Trigger SPU data processing on a limit event	<b>PACTR.TRIG</b>

## Signal Processing Unit (SPU)

**Table 601 Partial-acquisition Counter Specific Parameters for ADC Data (cont'd)**

Parameter	Definition	Comments
Error	Trigger an error interrupt on a limit event (unless TRIG bit is set and SPU is configured and waiting for data i.e. CTRL register written with MODE=INT)	<b>PACTR.ERR</b>
Attention	Trigger an attention request interrupt on a limit event	<b>PACTR.ATTN</b>
Counter Value	Current state of the Partial Acquisition Counter	<b>PACTR.COUNT</b>

Example use case for a measurement cycle with 1024 ramps of 256 samples and 4 antennae with the intention to process the first four ramps, skip two ramps while reprogramming the SPU and then process the remaining 1018 ramps:

- Program SPU for data from RIF, 256 samples and 4 antennae and four ramps, reset the partial-acquisition counter. Set the PACTR Error Interrupt and configure the limit field to 6 (as we expect to have reprogrammed the SPU by the start of the seventh ramp). Trigger the SPU by either internal (CTRL.MODE=INT) or external mode (CTRL.MODE=EXT).
  - This will initialise the Partial-acquisition counter with the correct information to keep track of the ramps
  - The SPU will process 4 ramps and then stop. The SPU DONE interrupt will then trigger the CPU to process the data. The CPU can then take decisions on how to best configure the SPU for the remainder of the measurement cycle
  - The Partial-acquisition counter will continue to track ramps. If the ramp counter reaches six (i.e. the seventh ramp has started) without the PACTR register being updated by the CPU, then the error interrupt will be asserted. This would indicate that the expected reconfiguration time has been exceeded.
- The CPU reprograms the SPU for the remainder of the measurement cycle<sup>1)</sup>. The partial-acquisition counter trigger bit is set and the error bit is set. The CTRL.MODE bitfield must be set to INT or EXT<sup>2)</sup>. The partial-acquisition counter is not reset.
  - The SPU is triggered at the start of ramp seven and the data from the seventh ramp to the one thousand and twenty-fourth ramp is acquired and processed. This trigger must be an internal trigger.
  - Setting the error bit eliminates the race condition of the start of the seventh ramp occurring between the write to the PACTR register and the write to the CTRL register. The error interrupt will be gated if TRIG is set and the CTRL register has been written with MODE=INT or with MODE=EXT and an external trigger has occurred.

The partial-acquisition counter can only be used to monitor input data sourced from the Radar Interface. It cannot be used to track data loaded from Radar Memory.

### Notes

1. *The SPU can process data from the Radar Memory without impacting the operation of the Partial Acquisition Counter once it is configured and running*

- 1) The reconfiguration of the SPU is not expected to change any setting in the ID\_CONF or ID\_CONF2 registers except the ID\_CONF.RAMPS field as the RIF(s) are still running the same measurement cycle and the format of the data input should not have changed. This restriction means that, although BYPASS could be used, reloading bypassed data before the end of the measurement cycle (i.e. in the gaps between partial acquisitions) is not supported.
- 2) If the partial-acquisition Counter trigger condition is reached before the CTRL register is programmed to enable the SPU, then an error interrupt will always be generated.

## Signal Processing Unit (SPU)

2. Setting the partial acquisition trigger (*PACTR.TRIG = 1*) without enabling the counter (*PACTR.EN = 0*) will still enable the trigger logic. Incoming Data will be filtered indefinitely waiting for a limit event which will not occur

### 19.3.1.4 FFT Data to Antenna Mapping

Some processing units (e.g. the complex windowing function) require that each set of FFT data be associated with an antenna. This is done using the information stored in the ID\_RM\_CONF.PM and ID\_RM\_CONF.AM bitfields.

If ID\_RM\_CONF.PM is set to  $1_B$ , then the buffer memory datablock will contain one dataset from each antenna. The processing units will therefore use the dataset index to identify the antenna.

If ID\_RM\_CONF.PM is set to  $0_B$ , then the data block is assumed to contain data from a single antenna. In this case, the setting of ID\_RM\_CONF.AM is used to define which of the loop repeat counters (Inner Loop Repeat or Outer Loop Repeat) is used to generate the antenna ID. Alternatively, the antenna ID can be forced to either  $0_D$  or “dataset index” (equivalent to ID\_RM\_CONF.PM= $1_B$ ). If the antenna ID is derived from one of the loop counters, only the three LSBs of the counter will be used to keep the antenna ID within the permitted range of  $0_D$  to  $7_D$ .

The loop repeat counter value used is chosen at the point the buffer memory is filled and before the counter is incremented for reading the next data. This leads to the following effects:

- If ID\_RM\_CONF.AM is set to BLR, the antenna ID will always be the three LSBs of the BLR value in the RM\_BLR register.
- If ID\_RM\_CONF.AM is set to ILR, the antenna ID will be as follows
  - for the final data block before incrementing the Outer Loop Counter, the antenna ID will be the 3 LSBs of the ILR value in RM\_IOLR
  - for other data blocks, the antenna ID will alternate between  $3_D$  and  $7_D$  for 32 bit precision complex data and will always be  $7_D$  for 16 bit precision data (real or complex format) or power data.
- If ID\_RM\_CONF.AM is set to OLR, the antenna ID will start at  $0_D$  and will then increment every time the Outer Loop Counter is incremented. If ID\_RM\_CONF.BLOCKS is not equal to zero, the value will be the final value of the Outer Loop Counter when the IDM completes loading of the buffer memory

### 19.3.1.5 Reading Power Data From Radar Memory

The Input DMA will normally read complex data from the Radar Memory as this is the expected format of FFT results. However, in some cases, further analysis of linear power data stored by a previous processing pass of the SPU may be desirable. As this data will be stored as a 32 bit, unsigned integer, ID\_RM\_CONF.FORMAT has an option for reading 32 bit power.

This setting is intended to allow power data to be processed by the MATH2 units and must only be used with the FFT accelerator and MATH1 complex data specific functions bypassed or disabled.

### 19.3.1.6 Reading 16 bit Real Data From Radar Memory

The Input DMA will normally read complex data from the Radar Memory as this is the expected format of FFT results. However, in some case, memory restrictions may require data to be stored as 16 bit precision, real only data. Reading 16 bit precision, real data can be done by setting ID\_RM\_CONF.FORMAT to REAL16BIT for integer data or REAL16FP for half precision floating point. If either of these options is used, then address transpose mode is not available and ID\_RM\_CONF.TRNSPS must be set to  $0_B$ .

For half precision floating point, denormalised numbers, Infinities and NaNs should be avoided in the input data. Denormalised numbers will be treated as zero. Infinities and NaNs will be set to either maximum negative or maximum positive values depending on the sign bit to allow processing to continue and will not propagate. The SPU has no capability to process signalling NaNs.

## Signal Processing Unit (SPU)

### 19.3.1.6.1 Buffer Memory Switching for 16 bit real data

If the ID\_RM\_CONF.PM bit is set to IM, this signals that the data block should contain one dataset for each antenna (i.e. the Inner Loop Offset is set to step across antennae) and the Buffer Memory will be switched when the Inner Loop Repeat Count is reached. Otherwise, the Input Buffer Memory will be switched every eight iterations of the Inner Loop or when all data has been read whichever occurs first.

### 19.3.1.7 Data Storage in Buffer Memory

The datasets will be stored linearly in the buffer memory in the order they should be processed. This allows the Data Loader module to progress sequentially through the buffer memory address space.

## 19.3.2 Streaming Processor 1

Streaming Processor 1 retrieves data from the buffer memory loaded by the Input DMA Engine and processes the information for use by the FFT accelerator

### 19.3.2.1 Double Pass Mode

The SPU allows all processing operations to be run twice per data block. This mode is enabled using the DPASS\_CONF.EN bitfield. Once the data has been loaded to Buffer RAM by the Input DMA Engine, the subsequent modules in the processing pipeline have the option to

- read the data twice and apply different processing options. This is the default “double pass” mode
- independently apply the different parameter sets defined for the window function to successive sequences of data blocks. See [Section 19.3.2.1.2](#) below.

This results in two sets of results being written to Radar Memory.

**Table 602 Parameters Specific to Double Pass Mode**

Parameter	Definition	Comments
Enable Mode	Configure SPU to process each data block twice.	<b>DPASS_CONF.EN</b>
Switch Mode	Configure SPU to process successive data blocks with different parameter sets. Number of data blocks between parameter switches configured by DPASS_CONF.COUNT	<b>DPASS_CONF.SWITCH</b>
Window Parameter Switch Count	Switch the window parameters when this many data blocks have been processed	<b>DPASS_CONF.COUNT</b>
Window Parameter Switch Count Enable	Enable the Window Parameter switch count	<b>DPASS_CONF.EN_CNT</b>

The values to be used in the second processing pass are stored in the BE1\_<REGNAME> registers. Primarily these are the settings for the MATH1 unit but the base address settings for the various data options in the MATH2 units are also duplicated.

The following register fields should not be changed between processing passes. A write to these fields in either copy of the register will be reflected in both registers.

## Signal Processing Unit (SPU)

**Table 603 Mirrored Register Fields**

Register	Field
LDR_CONF	DRPL
	DRPF
	EXPNT
	SIZE
	FFTBYPS
	IFFT
LDR_CONF2	PADF
UNLDR_CONF	FORMAT
	EXPNT

Note: Note that the mode of the FFT module (FFT or IFFT) cannot be varied between passes so the LDR\_CONF.IFFT bit is mirrored.

### 19.3.2.1.1 Double Pass Switch Mode

The behaviour of Double Pass Mode can be modified by setting the DPASS\_CONF.SWITCH bitfield. In this case, each data block is only processed once and the different parameter settings are applied to consecutive data blocks.

### 19.3.2.1.2 Window Parameter Switch

The switch of the two possible sets of parameters for the window function can be made independent of the double pass functionality by setting the DPASS\_CONF.EN\_CNT. When this bit is set, the window parameters are selected independently of the double pass function and applied to sequences of data blocks. The number of data blocks per sequence is configured by the DPASS\_CONF.COUNT bitfield and is set to COUNT+1. If COUNT=0, then the different parameter settings are applied to alternate data blocks. If COUNT=7, then the different parameter settings are applied to sequences of 8 data blocks.

### 19.3.2.2 Data Loader Unit

The Data Loader will read data from memory and reformat it before pushing it into the MATH 1 unit pipeline. It can modify the format of the input data by resizing it from 16 to 32 bit precision and by converting it from real to complex format. Operations will occur in the following order

- 16 to 32 bit precision
- real to complex conversion

**Table 604 Loader Configuration Parameters**

Parameter	Definition	Comments
number of samples	number of samples per FFT currently in the buffer RAM	passed from the Input DMA Engine
number of data sets	number of FFTs to run	passed from the Input DMA Engine

## Signal Processing Unit (SPU)

**Table 604 Loader Configuration Parameters (cont'd)**

Parameter	Definition	Comments
Data precision	16 or 32 bits	passed from the Input DMA Engine
Data type	Real or complex	passed from the Input DMA Engine
Data format	Integer or floating point	applicable to Radar Memory data only. passed from the Input DMA Engine
Integration Mode Bandwidth Optimisation	Buffer Memory Configuration. Number of Datablocks in the buffer memory	passed from the Input DMA Engine
Signed or Unsigned	Input Data is unsigned or 2's complement format	Applicable to ADC data only, passed from the Input DMA Engine

### 19.3.2.2.1 Data Reformatting

The format of the data in the buffer memory is known as it is programmed into the Input DMA Engine configuration. All data processed by the FFT accelerator must be in 32 bit complex format. Real data is converted to complex data with a zero imaginary component. 16 bit data is extended to 32 bit using the method specified by the LDR\_CONF.EXPNT bitfield. This controls how the 16 bit data is justified within the 32 bit output field. The type of data (signed or unsigned) is passed from the Input DMA Engine and is derived from the ID\_CONF.SIGNED bitfield. This determines whether a signed or unsigned extension to 32 bits is used.

When using data from Radar Memory, LDR\_CONF.EXPNT has a maximum permissible value of  $16_D$  and aligns 16 bit fixed exponent format data. When using data from the Radar Interface, the number of significant bits can be less than  $16_D$  so the maximum permissible value of LDR\_CONF.EXPNT can be increased to allow for 10 bit data right aligned in the 16 bit field.

If the shift operation causes a loss of significant data bits, then the output of the shift operation will saturate at full scale.

If the input data format is complex, then both real and imaginary components are treated in the same manner.

### 19.3.2.2.2 Integration Mode Bandwidth Optimisation

This is described in [Chapter 19.3.1.2.6, “Bandwidth Optimisation Integration Processing Mode.” on Page 43](#).

The Integration Mode Bandwidth Optimisation of the Input DMA Engine results in multiple datablocks being present in the buffer memory. The Buffer memory would normally be switched between datablocks. When this mode is enabled, instead of switching the buffer memory, the buffer memory address is offset to point to the next datablock. The MATH1, FFT and all units downstream of this point therefore see separate datablocks as normal.

### 19.3.2.2.3 Overview of Data Truncation and Padding

The diagram below gives a functional overview of how the operations described below affect the bins of the data loaded from the buffer memory

## Signal Processing Unit (SPU)

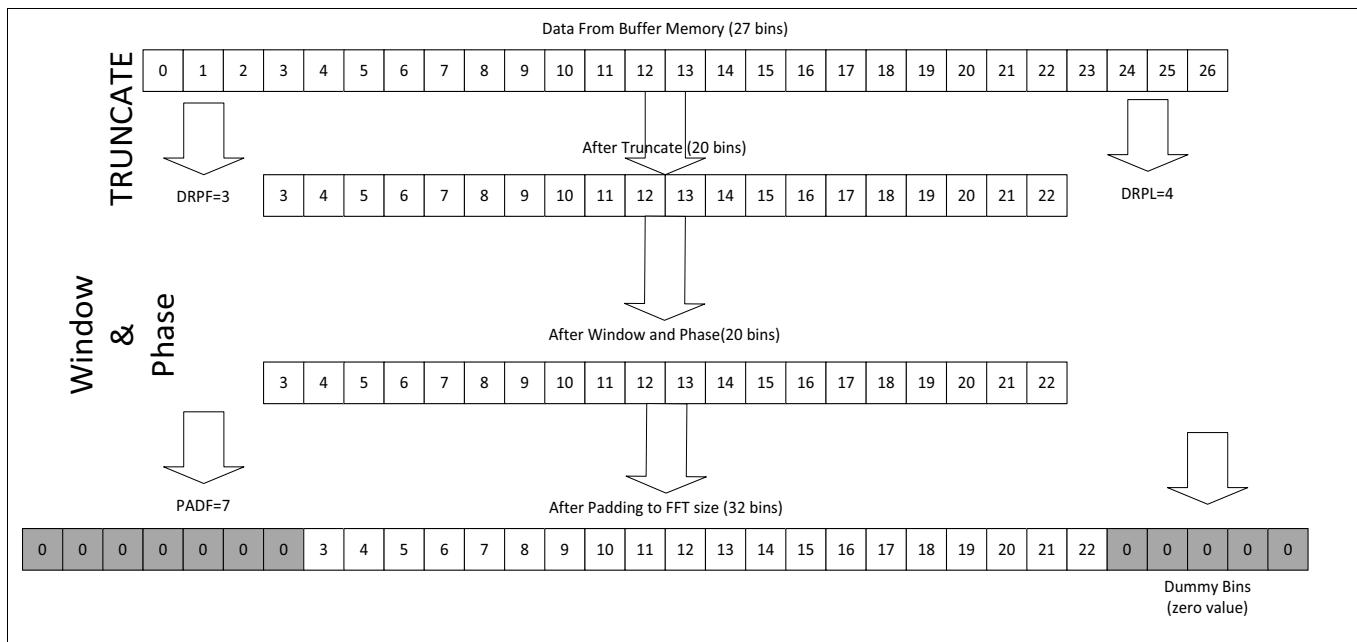


Figure 249 MATH1 Operation Overview

### 19.3.2.3 MATH1 Unit

The MATH1 unit handles all transforms applied to the data before the FFT processor. The operations will be applied in the following order

- Truncation
- Windowing
- Phase Shift
- Padding

Table 605 MATH1 Configuration Parameters

Parameter	Definition	Comments
Number of samples	Number of samples per FFT to be loaded into the FFT accelerator	<b>LDR_CONF.SIZE</b>
Number of data sets	Number of FFTs to run	passed from the Loader Unit
Window base address	Base address in Config Memory where the windowing functions are stored	<b>LDR_CONF2.WBASE</b>
Antenna window offset address	Offset from the Window base address of the window function for a particular antenna	<b>Ax_ANTFOST.ADROFSTy</b> <b>where x=0-3,y=0-1</b>
Padding activation	Padding enable / disable	implicit activation when the FFT size from the loader does not match the FFT size for the FFT accelerator
Leading padding	position of the 1st operand in the FFT (others being zeroed).	<b>LDR_CONF2.PADF</b>
Fast phase shift	does a 0/90/180/270 phase shift without using the windowing memory	<b>LDR_CONF.PHSHFT</b>

## Signal Processing Unit (SPU)

### 19.3.2.3.1 Truncation

Truncation is performed by two register fields, LDR\_CONF.DRPF and LDR\_CONF.DRPL. The reference for the dataset size is taken from the Input DMA Engine Configuration, IDM\_CONF.SMPLCNT field. DRPF is used to truncate from the beginning of the dataset and DRPL to truncate from the end of the dataset.

Truncation is not supported if the FFT is bypassed (LDR\_CONF.FFT\_BYPS = 1<sub>B</sub>) and both DRPF and DRPL must be set to 0<sub>D</sub> in this case. Behaviour if this restriction is not observed is undefined.

### 19.3.2.3.2 Windowing

Windowing is enable using the LDR\_CONF2.WINEN bit. The coefficients for the windowing function are stored in the configuration memory. The output of the windowing unit is scaled so that the maximum value of the window coefficients is +(2<sup>31</sup>-1)/2<sup>31</sup> and the minimum value is -1.

To retain precision, any needed truncation will be done at the output of arithmetic operations rather than the input. Where a result needs to be scaled down to fit into an output, the result will be rounded down to nearest integer after division.

The base address of the window coefficients is stored in LDR\_CONF2.WBASE. Additionally an address offset can be defined for each antenna. The offset is defined in bytes. If the address offsets are set to zero, the same window will be used for all antennae. Offsets are stored in the ANTOFST fields of the A[0|1]\_ANTOFST registers in the following order:

**Table 606 Antenna to Offset Register Mapping**

Antenna	Register	Bitfield
0	A0_ANTOFST	ADROFST0_ANT
1	A0_ANTOFST	ADROFST1_ANT
2	A1_ANTOFST	ADROFST0_ANT
3	A1_ANTOFST	ADROFST1_ANT
4	A2_ANTOFST	ADROFST0_ANT
5	A2_ANTOFST	ADROFST1_ANT
6	A3_ANTOFST	ADROFST0_ANT
7	A3_ANTOFST	ADROFST1_ANT

Antenna number selection is controlled by the ID\_RM\_CONF.AM and ID\_RM\_CONF.PM bitfields.

The window size must be the same as the truncated data set (i.e. before the padding operation).

The window data can be stored in several formats to save memory. Maximum precision is achieved using 32 bit complex window coefficients. However the data can also be stored in 32 bit real format, in which case the windowing function can affect magnitude but not phase. Further compression can be achieved by using 16 bit formats. Format is set using the LDR\_CONF.FRMT bitfield.

Window coefficients stored as 16 bit are extended to 32 bit by extending with 16 LSBs set to 0<sub>B</sub> so that the stored coefficient is left justified in the 32 bit value.

### 19.3.2.3.3 Phase Shift

The phase shift is achieved by sign manipulation of the real and imaginary components of the complex data values as described in the following table.

## Signal Processing Unit (SPU)

**Table 607 Phase Shift Operation**

Shift (degrees)	LDR_CONF.PHSHFT	Output to Input Mapping	
		Real OP	Imaginary OP
0	00	(Real IP) * 1	(Imag IP) * 1
90	01	(Imag IP) * -1	(Real IP) * 1
180	10	(Real IP) * -1	(Imag IP) * -1
270	11	(Imag IP) * 1	(Real IP) * -1

### 19.3.2.3.4 Padding

The truncated dataset is then padded to the FFT size set by the LDR\_CONF.SIZE bitfield. The LDR\_CONF2.PADF bitfield is used to determine how many zero data points are added to the front of the dataset. The remainder are added at the end.

These LDR\_CONF2.PADF field has no effect if the FFT is bypassed (LDR\_CONF.FFT\_BYPS = 1<sub>B</sub>) and no padding is performed on the data in this case.

Note that the dataset size cannot exceed the size set in the LDR\_CONF.SIZE bitfield. If the programmed value of LDR\_CONF2.PADF would violate this limit, then the behaviour of the loader is undefined but correct results will not be obtained. Similarly, LDR\_CONF.SIZE must not be set to a smaller number than the number of data points left after the Truncate operation.

### 19.3.2.4 FFT Accelerator

The FFT accelerator uses a pipelined architecture to implement the efficient radix 2<sup>2</sup> architecture.

The FFT accelerator can be completely bypassed using the LDR\_CONF.FFTBYP bit.

When performing an FFT, the accelerator will divide the output by the number of data points in the FFT to ensure that the output data does not overflow the integer output format.

An inverse FFT can be performed by setting the LDR\_CONF.IFFT bit. Setting the mode to IFFT will disable the descaling function of the FFT accelerator IP and the output will not be divided.

**Note:** *The inverse FFT function is intended to be used to post-process FFT results to recreate the original input data. If arbitrary input data is used or windowing or scaling is applied to the FFT results before attempting an inverse FFT, an overflow may occur at the output of the FFT accelerator which would normally be compensated for by the descaling operation. This overflow will be signalled to the processor by interrupt if the STAT.INTMSK[1] bit is set.*

Note that the dataset size cannot exceed the size set in the LDR\_CONF.SIZE bitfield. The number of datapoints read from the Buffer RAM is determined by the Input DMA configuration. If the programmed values of LDR\_CONF.DRPF, LDR\_CONF.DRPL and LDR\_CONF2.PADF results in a number of datapoints that violate this limit, then the behaviour of the loader is undefined but correct results will not be obtained.

**Table 608 FFT Engine Configuration Parameters**

Parameter	Definition	Comments
FFT length	defined in the global registers as the number of operands for the inner loop	<b>LDR_CONF.SIZE</b>
counter of actual FFT loads	counter. reset by user software	safety requirement

## Signal Processing Unit (SPU)

**Table 608 FFT Engine Configuration Parameters (cont'd)**

Parameter	Definition	Comments
counter of actual FFT unloads	counter. reset by user software.	safety requirement
FFT enable	FFT enabling / disabling	logically the same as “not FFT bypass” <b>LDR_CONF.FFT_BYPS</b>
Inverse FFT Mode	Configure the FFT accelerator to perform and Inverse FFT operation instead of the normal FFT	<b>LDR_CONF.IFFT</b>

### 19.3.3 Data Unloader

The Data Unloader writes the data from the FFT accelerator module into the output buffer RAM. All user programmable options are associated with the sideband processing of signal power to generate a histogram. By default, the Data Unloader writes 32 bit precision complex data to the Buffer RAM. As a space saving measure, the UNLDR\_CONF.FORMAT bitfield can be set to 16BIT and the UNLDR\_CONF.EXPNT bitfield used to set a common exponent value for the written data. The 32 bit values will then be shifted right by the number of bit positions programmed in the UNLDR\_CONF.EXPNT field before a saturating truncation to the 16 LSBs is performed (see “[Saturating Truncation” on Page 61](#) for further details). When using this mode, the value of the EXPNT should be restricted to values between 0 and 16. The behaviour of the Data Unloader with values outside this range is undefined.

If writing of 32 bit precision data is configured, then setting the EXPNT field to a non-zero value will cause the FFT output data to be shifted left by the number of bits set in the EXNPT field (preserving sign). If this causes the data value to overflow, then the data will be set to full scale minimum or maximum value depending on the sign.

*Note:* *If the data being processed is 32 bit precision linear power (ID\_RM\_CONF FORMAT=PWR32BIT, then compression is not supported and the data format (UNLDR\_CONF FORMAT) should always be set to 32BIT.*

The Data Unloader also tracks the number of sign bits (i.e. leading ones and zeros) in the FFT output data for use by the application software.

**Table 609 Unloader Configuration Parameters**

Parameter	Definition	Comments
Common Exponent	The 32 bit data will be shifted right by this number of bits before truncation to the 16 LSBs if the FORMAT selected is 16 bit and left if the format is 32 bit	<b>UNLDR_CONF.EXPNT</b>
Format	Select the format of the data to be written to the buffer memory. Options are 16 or 32 bit precision complex.	<b>UNLDR_CONF FORMAT</b>

#### 19.3.3.1 Power Histogram

The power histogram module stores a histogram of the FFT output power distribution into the configuration RAM. The method used derives the histogram bin index from the  $\log_2$  of the power. This method first requires the linear power to be calculated.

## Signal Processing Unit (SPU)

The linear power value used is derived in two possible ways:

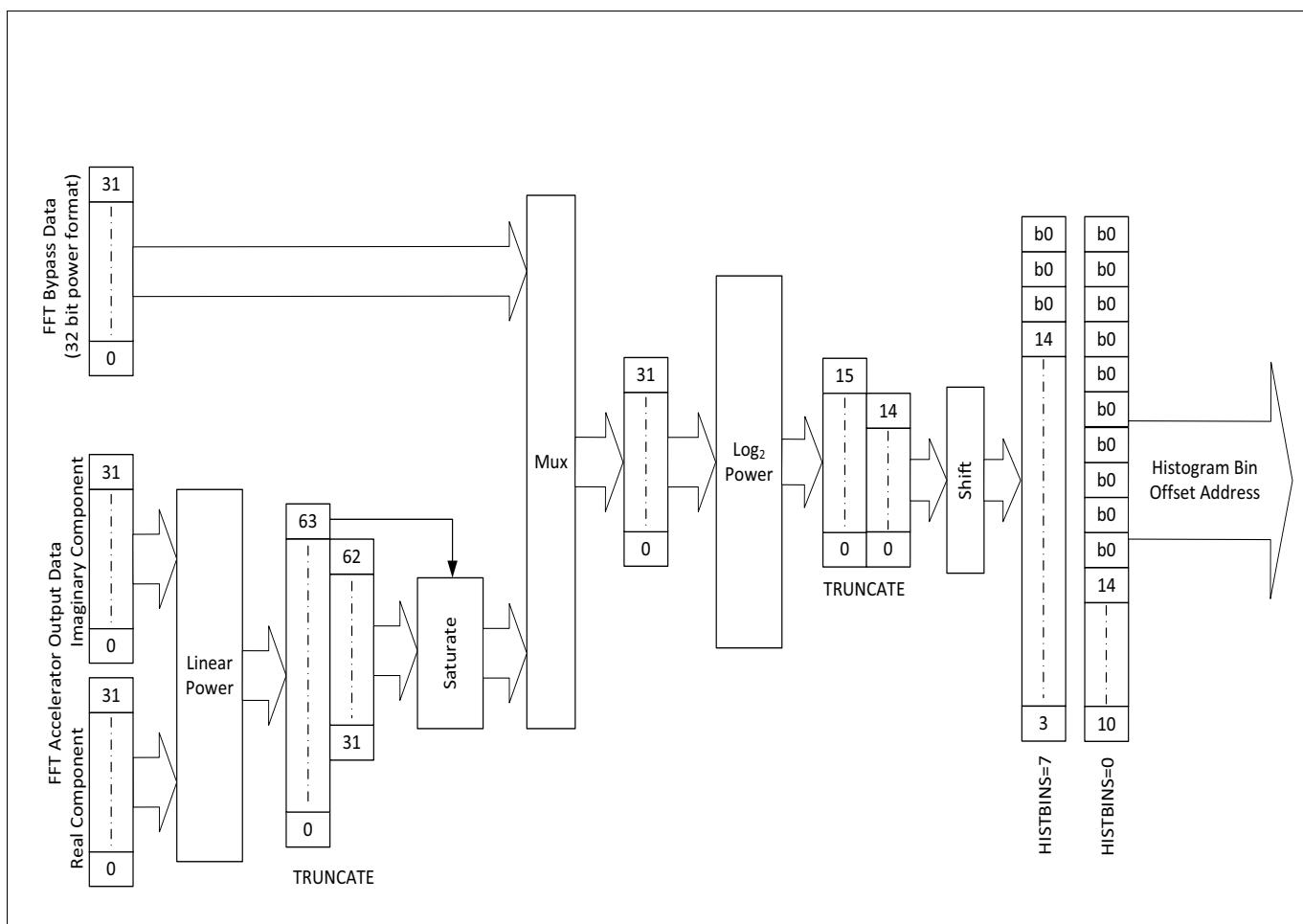
- If the data format is complex, it will be calculated to 32 bit precision using the method described in “[Pre-Processing Units” on Page 60](#). This method ensures that all 32 bits of the calculated power value contain useful data (i.e. all bits will toggle, no bits will be at a constant value over the full range of inputs)
- If the data format is 32 bit power, the data will be used directly.

**Note:** *The value of the ODP\_CONF.FTR bitfield is considered when calculating the power values used for the histogram. If ODP\_CONF.FTR is set, then the imaginary component of the FFT output data will be set to 0<sub>D</sub>.*

This 32 bit linear power value is used to derive a 16 bit log<sub>2</sub> power value. The log<sub>2</sub> power value is formatted as:

- bit 15: sign bit, always 0 (inputs of less than 2<sup>0</sup> are mapped to a result of 0)
- bits 14 downto 10: integer part
- bits 9 downto 0: fractional part

The log<sub>2</sub> power value is then right shifted by a programmable number of bits and used as a word address offset from the histogram base address in the configuration memory. The 64 bit word stored at this address is incremented. As bit 15 is always zero, it is specifically excluded from the address



**Figure 250 Diagrammatic Representation of Histogram Bin Offset Generation**

The histogram is enabled by writing an 1<sub>B</sub> to UNLDR\_CONF.HIST\_EN, the base address is stored in UNLDR\_CONF.HISTBASE. The number of bins is configured by the value in UNLDR\_CONF.HISTBINS. A minimum

## Signal Processing Unit (SPU)

of 32 bins and a maximum of 4096 bins are supported. The number of bins must always be a power of 2 and is given by  $2^{(\text{HISTBINS}+5)}$ . The size of the right shift referred to in the previous paragraph is therefore (10-HISTBINS).

The byte address of any given bin is UNLDR\_CONF.HISTBASE + ((BIN NR.) \* 8).

The power histogram will be calculated for the number of FFTs controlled by the values in the UNLDR\_CONF2.START and UNLDR\_CONF2.END fields. The FFTs of the measurement cycle are counted with the first FFT being FFT 0. If the count is greater than or equal to UNLDR\_CONF2.START and less than or equal to UNLDR\_CONF2.END, then the FFT is used to compute the histogram. If UNLDR\_CONF2.END is set to all 1, then accumulation of histogram data will run continuously from when the count reaches the UNLDR\_CONF2.START value until the end of the measurement cycle.

The calculation of the power histogram can be further filtered by considering antenna ID. This mode is enabled by setting UNLDR\_CONF.HAFE. If set, only FFT results associated with the antenna ID stored in the UNLDR\_CONF.AVF will contribute to the power histogram or update the FFT count used for checks against UNLDR\_CONF2.START and UNLDR\_CONF2.END values.

The histogram needs two clock cycles to process each bin as it needs to read the current value from the configuration memory and then write it back after updating. Consequently, the histogram must only be enabled if the FFT clock rate (CTRL.DIV) is set to something other than UNITY.

**Note:** *The histogram and the windowing function in the LOADER both require full bandwidth access to the Configuration RAM while the SPU is running. To prevent conflict, the Configuration Memory is split into two towers. One tower is addressed from offset 0 to CONFIG\_RAM\_SIZE/2-1 and the other tower from offset CONFIG\_RAM\_SIZE/2 to CONFIG\_RAM\_SIZE-1. The memory used for histogram and the memory used for the window functions must not be in the same tower if both functions are active at the same time. The memory allocated from the histogram must not be split between both towers under any circumstances as this is not supported by the hardware implementation.*

**Table 610 Histogram Configuration Parameters**

Parameter	Definition	Comments
Histogram enable	enable / disable for histogram	<b>UNLDR_CONF.HISTEN</b>
Number of classes	from 32 to 4096 classes	<b>UNLDR_CONF.HISTBINS</b>
Base address of the histogram in the RAM		note that we may have a modulo here to avoid too high complexity in address computations <b>UNLDR_CONF.HISTBASE</b>
Start Delay	FFTs to count before starting histogram calculation	<b>UNLDR_CONF2.START</b>
End Delay	FFTs to count before ending histogram calculation	<b>UNLDR_CONF2.END</b>
Filter by Antennae	Enable histogram data to be accumulated only for a single antenna	<b>UNLDR_CONF.HAFE</b>
Antenna Number to use for filtering	Index of Antenna to be used for histogram data accumulation	<b>UNLDR_CONF.AFE</b>
Real Data Format	Ignore imaginary component when computing power	<b>ODP_CONF.FTR</b>

## Signal Processing Unit (SPU)

### 19.3.3.2 Statistical Information

The Data Unloader can calculate the mean value of the power for each FFT while processing the data. The information will be stored in a FIFO for later use by the Output Data Processor. Calculation and storage of this information is enabled using the SBCTRL.EN bitfield.

The mean is always calculated from 32 bit data. If 16 bit complex format is selected by UNLDR\_CONF.FORMAT, the mean calculation will use the 32 bit data before the shift defined by UNLDR\_CONF.EXPNT is applied. If 32 bit complex format is selected, the mean calculation will use the 32 bit data after the shift defined in UNLDR\_CONF.EXPNT is applied. This ensures that the mean is calculated using the same data as used by the Output Data Processor.

The linear power value used is calculated using the method described in “[Pre-Processing Units” on Page 60](#).

This gives us half the information needed to compute the variance (square of standard deviation). To complete the variance calculation, a second pass through the data is needed. This will be done when the data is read by the Output Data Processor.

Statistical information for the measurement cycle can be stored in the Radar Memory. One set each of minimum power, maximum power, mean power and standard variance will be stored for each FFT processed.

The division used to complete the variance<sup>1)</sup> calculation is a “shift right” operation rather than a full hardware division. This causes some inaccuracies when the number of samples is not a power of two. This does not affect the statistical information computed from FFT results but does mean that calculation of the statistical information when the FFT engine is bypassed (LDR\_CONF.FFT\_BYPS = 1<sub>B</sub>) uses a mean which effectively pads the number of samples to the next power of two with zero values.

These operations are described in more detail in the relevant section in the Output Data Processor section.

**Note:** *To ensure that the data used for the statistical calculations is consistent the value of the ODP\_CONF.FTR bitfield is considered when calculating the power values used for the computing the mean. If ODP\_CONF.FTR is set, then the imaginary component of the FFT output data will be set to 0<sub>D</sub>.*

### 19.3.4 Streaming Processor 2, The Output Data Processor

The Output Data Processor module formats and writes the results datasets to the Radar Memory. It can also route the data through multiple processing modules working on complex data and power. The Output Data Processor is split into two submodules, the MATH2 Unit and the Output DMA Engine. When reading data from the Buffer RAM, the Processor uses the UNLDR\_CONF.FORMAT and UNLDR\_CONF.EXPNT to determine whether the stored data is in native 32 bit precision format or has been compressed to 16 bit precision and requires reformatting.

The ODP has a processing path for manipulating the base data and bin rejection working in parallel with multiple analysis units (CFAR, NCI, statistics and arithmetic).

#### 19.3.4.1 Streaming Processor 2 Data Fetch from Buffer Memory

There are two classes of processing path through the MATH2 which require the data to be read from Buffer RAM in two incompatible ways.

- Operations such as the FFT data output path through the bin rejection unit require the data to be read as consecutive points from each FFT result.
- Operations using the data integrated across antenna requires equivalent data points to be read consecutively from each FFT result stored in the buffer memory so the integrated value can be calculated.

<sup>1)</sup> The variance calculated is the population variance rather than a sample variance. The sum of the squares is divided by the number of samples rather than the number of samples minus one.

## Signal Processing Unit (SPU)

If both types of path are enabled by the programmed configuration options, then this conflict is addressed by reading the data from the buffer memory twice. The first fetch will be used for the operations requiring linear addressing by FFT result. The second fetch will be performed for operations requiring the integrated result across antennae. The number of fetches required is evaluated and executed independently for each pass if double pass mode is enabled.

### 19.3.4.2 In Place FFT

Setting ODP\_CONF.IPF configures the Output Data Processor to generate addresses using the same scheme as the Input DMA uses to read the data being processed. The intention is that the output data will be written to the memory locations cleared by the reading of the input data.

To do this the Output DMA uses an equivalent set of counter and offset registers to those instantiated in the Input DMA engine. The parameters used to configure this duplicate register set are inherited from the Input DMA configuration registers. This causes the address sequence generated by the Output DMA to exactly duplicate the address sequence used by the Input DMA to read the data. Addresses in this case refer to the 256 bit words of the EMEM. The Output DMA will always assemble enough results to completely fill each 256 bit word before moving to the next address in the sequence. This will result in fewer words being written than read if the output data size is smaller than the input data size. Conversely, if the output data requires more memory than the input data, results will be unpredictable and, consequently, use of in place FFT in these cases is not supported.

In Place FFT addressing only applies to the writing of the FFT results.

It is strongly recommended that bin rejection is not used with in place FFT when using transposed addressing as no additional memory saving will be possible (data will be written as a sparse array in memory) and addressing of the resultant data will be extremely complex.

#### 19.3.4.2.1 Restrictions

Correct operation of this mode requires that the Input DMA reads and uses all 32 bytes of each word in Radar Memory in a single pass. If only part of the word is used, it is probable that the Output Data Processor will overwrite the location before a further pass through the data cube by the Input DMA uses the rest of the word.

This mode cannot be used if Double Pass mode is used to write two sets of FFT results.

### 19.3.5 MATH2 Unit

The MATH2 Unit implements multiple parallel processing units as shown in [Figure 251](#) below. These processing units are grouped into two data flows, one working on the raw complex data and the other on linear or  $\log_2$  power. The following table shows the operating domain and concurrency of the various functions:

**Table 611 MATH2 Unit Functional Concurrency**

	Function - Only one function can be active for each O/P Channel										
	Stats	Local Max	CFAR CA	CFAR GOS	FFT Data <sup>1)</sup>	Log <sub>2</sub> Signal Power	Vector Add/ NCI <sup>2)</sup>	Vector Add/ CI <sup>2)3)</sup>	Sum-LinP	Sum	Sum Log <sub>2</sub> Power
Power Domain <sup>4)</sup>	Yes	Yes	Yes		No	Yes	Yes	No	Yes <sup>5)</sup>	No	Yes
Complex Domain <sup>6)</sup>	No	No			Yes	No	No	Yes <sup>7)</sup>	No	Yes	No
O/P Channel	PORT3	PORT4		PORT5 <sup>8)</sup>	PORT1	PORT2	PORT6	PORT7			PORT8

## Signal Processing Unit (SPU)

- 1) Incorporates any or all of scalar add, scalar multiply and bin rejection.
- 2) Running these functions simultaneously will be bandwidth limited
- 3) Sum Antenna on Complex Data
- 4) Function is possible using power as input
- 5) Sum of the linear power
- 6) Function is possible with Complex Values as Input
- 7) Vector add on the FFT result complex data is done only across antenna.
- 8) Using the CFAR GOS engine also blocks Local Max on Port 4

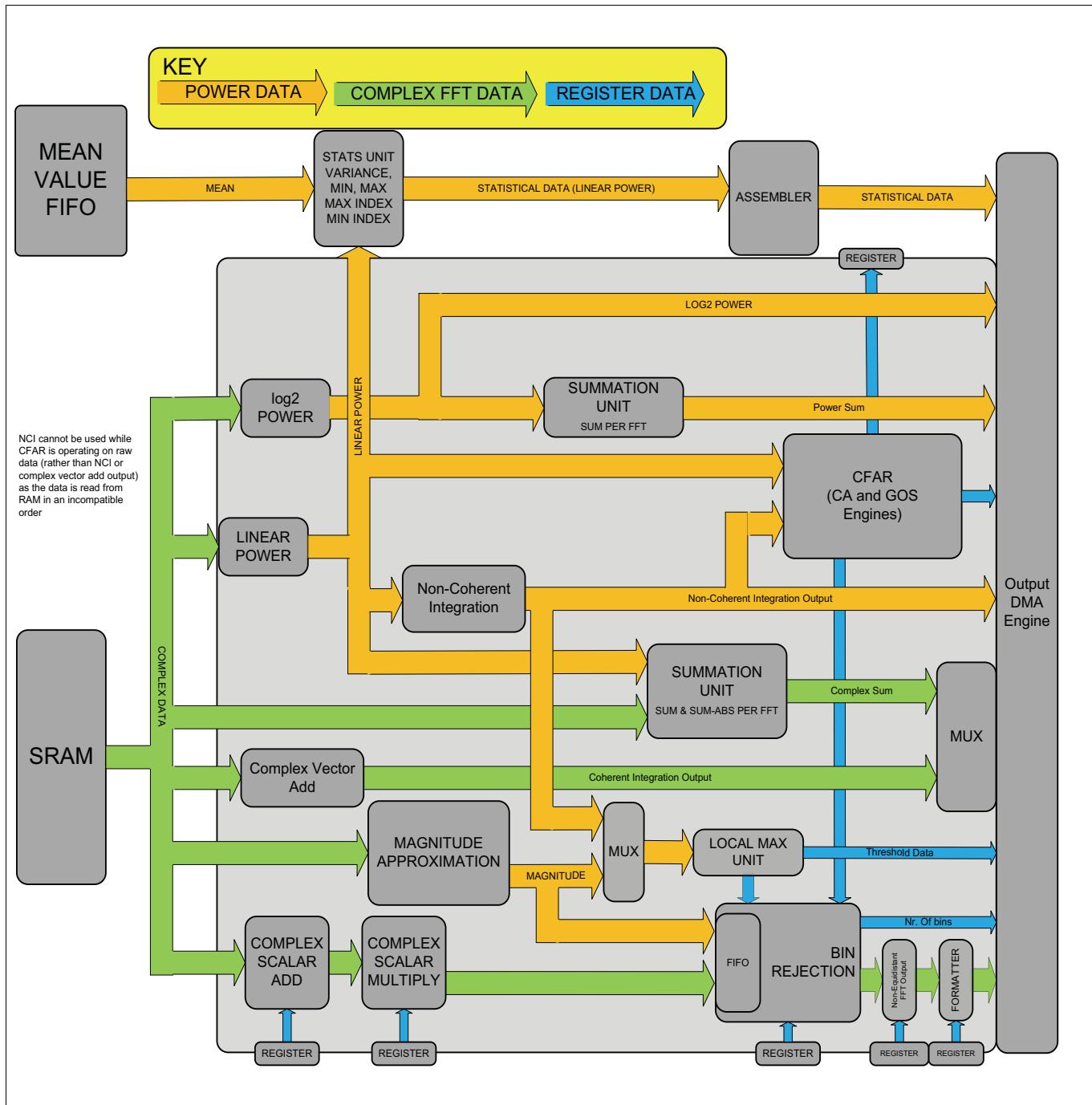
For expected performance using the MATH2 unit, refer to [Chapter 19.7.14.4, “MATH2/ODM Performance”](#)

**Table 612 Output DMA Engine Data Streams**

Output port	Definition	Comments	Precision. (Bits)	Padding <sup>1)</sup> per FFT or measurement cycle
Port1	3 sources of informationFFT unitFFT BIN rejection and zeroingadd scalar or complex to vector		16 or 32 bit precision, complex or real formats	FFT
Port2	$\log_2$ signal power		16 bits	FFT
Port3	min, max, average, (standard deviation) <sup>2)</sup>		32 bits	FFT
	index of min, index of max	12 bits of data, padded to 16 bits with leading zeroes	16 bit	
Port4	local max or CFAR (CA Engine)		1024 bits max	cycle
Port5	CFAR (GOS Engine)		1024 bits max	cycle
Port6	Vector add /sum, power domain	typically: non coherent integration	32 bits	FFT
Port 7	sum, sum-linp	summation over each FFT	32 bits	cycle
	sum-antenna	typically, sum for data preparation before elevation computations <sup>2)</sup>	16 or 32 bit complex	FFT
Port 8	Sum of Log <sub>2</sub> Power	Sum of log <sub>2</sub> power over each FFT	16 bits	cycle

- 1) Padding is used to ensure that the next set of data written starts on a 256 bit aligned address. Data intended to be read back into the SPU will have one element per FFT bin processed and each dataset must start on a 256 bit aligned address. These data streams will be padded per FFT. Other data streams will be padded per measurement cycle to optimise memory usage
- 2) Output from this operation is a 1D vector of the same length as the input FFTs

## Signal Processing Unit (SPU)



**Figure 251 MATH2 Unit Architecture**

The MATH2 unit will always expand the data read from the buffer memory to 32 bit precision before using it in any functions. The expansion function is the inverse of the compression operation used in the Unloader with any additional LSBs being padded with  $0_B$  and the MSBs being sign extended. See [Section 19.3.3, “Data Unloader” on Page 53](#).

Where 32 bit precision values are multiplied to generate a result at 32 bit precision, rescaling of the result will take place after the multiplication operation. Rescaling will always remove the least significant bits of the result.

When multiplying two signed 32 bit numbers, the result will be a 64 bit signed value and this will be scaled to 32 bits by dividing by  $2^{31}$  and rounding down to the nearest integer while discarding the MSB. As multiplying two full scale negative numbers will result in an overflow using this algorithm, this corner case will generate a full scale, positive result.

## Signal Processing Unit (SPU)

When multiplying two unsigned 32 bit numbers, the result will be a 64 bit unsigned value and this will be scaled to 32 bits by dividing by  $2^{32}$  and rounding down to the nearest integer.

If any calculation result is scaled to 16 bits, then the result will be rounded to the nearest integer after division. Results which fall exactly midway between two integers will be rounded away from zero.

Scaling of the results of 32 bit addition operations to produce a 32 bit output are handled in a similar manner. The intermediate result is always sized so that overflow does not occur and then this is scaled down to 32 bits by dividing and rounding down towards -Infinity to the nearest integer. The scaling to 32 bits will, in most cases, occur automatically unless the number of operands is configurable, in which case, a configurable scaling factor has been provided.

### 19.3.5.1 Pre-Processing Units

The pre-processing units convert the raw FFT data to  $\log_2$  power, linear power or magnitude or change the precision of the data.

#### 19.3.5.1.1 Linear Power Calculation

Linear power is calculated to 32 bit precision by

- squaring the real and imaginary components and scaling by dividing by  $2^{31}$ .
  - For full scale positive data, this gives a value of  $7FFF\_FFFE_H$
  - For full scale negative data, this gives a value of  $8000\_0000_H$
- adding the truncated squares of the real and imaginary components and saturating the results
  - $7FFF\_FFFE_H$  for both real and imaginary complex components gives  $FFFF\_FFFC_H$  when the squares are added and saturation does not take place
  - $8000\_0000_H$  for both real and imaginary complex components gives  $1\_0000\_0000_H$  when the squares are added and  $FFFF\_FFFF_H$  after saturation.

#### 19.3.5.1.2 $\log_2$ Power Calculation

$\log_2$  power is computed to a precision of 16 bits and will be stored as an unsigned integer value with 7 integer bits and 9 fractional bits. This is a representation of the 32 bit precision linear power value scaled to 64 bits by multiplying by  $2^{31}$ . This is done to be compatible with the  $\log_2$  power calculation used inside the CFAR module which uses a 64 bit linear power input. An input linear power value of  $2^{31}$  scales to a 64 bit value of  $2^{62}$ . A 64 bit value of  $2^{62}$  corresponds to a  $\log_2$  value of  $7E00_H$ . The MSB of the 16 bit value is a sign bit.

The algorithm used is as follows:

- In the first stage, the location of the first bit set in the 64 bit input value is used to determine the integer component of the output.
- In the second phase of the algorithm, the most significant eight bits of the residual from the first phase is used as the index into a 256 entry look up table used to apply an appropriate fractional correction to the integer value

The maximum granularity of the look up table output is 0.006. An input value of  $0000\_0000_H$  is mapped to an output value of  $0000_H$ .

## Signal Processing Unit (SPU)

### 19.3.5.1.3 Magnitude Approximation

For magnitude, the approximation:

(19.1)

$$\text{MAGNITUDE} = \alpha \times (\text{MAX}(\text{ABS}(I), \text{ABS}(Q))) + \beta \times \text{MIN}(\text{ABS}(I), \text{ABS}(Q))$$

is used. The two multiplication constants are stored in the ALPHA and BETA bitfields of the MAGAPPROX register. The magnitude value is calculated at 32 bit precision. ALPHA and BETA are unsigned constants scaled to have a value range of between 0 and 1 approximately (actually 0.99998 to five significant digits). The output is a 32 bit value scaled to have an overall gain of 1.

**Note:** *The ABS function used in the magnitude approximation maps a full scale negative number  $8000\_0000_H$  to a magnitude of  $7FFF\_FFFF_H$*

Later operations then use the magnitude value. Operations which use magnitude data are:

- Threshold Detection, [Section 19.3.5.3.3](#)
- Local Maximum, [Section 19.3.5.2](#)

### 19.3.5.1.4 Saturating Truncation

The saturating truncation operation will remove the 16 MSBs of a 32 bit quantity to leave a 16 bit result. The intention is that the 32 bit quantity is scaled such that no significant information is lost when doing this (i.e. the 17 MSBs contain only sign information). However if this not the case and one or more of the MSBs contain  $1_B$  for positive quantities or  $0_B$  for negative quantities, then the 16 bit result will be set to full scale positive in the first case and full scale negative in the second case.

### 19.3.5.2 Local Maximum Detection Unit

The module checks to see if the bin being checked is a local maximum by comparing against adjacent bins. If it is, then the bin will be flagged. The module can also simultaneously check the bin against a threshold value. The output information can either be written to Radar Memory or used by the bin rejection unit. The unit therefore has two operating modes:

- In-line Mode. In this mode, the first FFT result of the measurement cycle<sup>1)</sup> is used to set the bins to be flagged for the entire measurement cycle. This information is passed to the Bin Rejection Unit. This mode requires additional time and buffer RAM read accesses to read the first FFT data for processing Local Maximum Unit. The number of extra reads and therefore the delay will be dependent on the size of the Local Max window.
- Off-line Mode. In this mode, each FFT result is used to generate a set of data. This data is then written to Radar Memory as a table.

This module cannot be used concurrently with the CFAR module. The Local Maximum Detection Unit has the same options for range and velocity spectrum extension as the CFAR module. See [Section 19.3.5.5.9, CFAR spectrum extension](#) for a full description.

**Note:** *The CFARCTRL.CFAREN bitfield should not be set to LCLMAXI, LCLMAXO or LCLMAXN without using setting LCLMAX.LMODE or LCLMAX.TMODE to a value other than “OFF” as this will affect performance and cause the Bin Rejection Unit to behave in an unpredictable manner.*

<sup>1)</sup> If double pass mode is enabled then the process of identifying datapoint of interest is repeated for the first FFT of the measurement cycle for each pass and both sets of information are stored. The appropriate set of data will then be used for processing all further FFTs of the relevant pass.

## Signal Processing Unit (SPU)

**Table 613 Local Maximum Configuration Parameters**

Parameter	Definition	Comments
local max enable	enable / disable	Combined with the control for the CFAR module as these are mutually exclusive <b>CFARCTRL.CFAREN</b>
Radar Memory Address	Base address to be used for writing the threshold results to the Radar Memory	<b>CFARCTRL.BASE</b>
Local Max Mode <sup>1)</sup>	Off, Reject if not local max, reject if local max	<b>LCLMAX.LMODE</b>
Local Max Check Combining	Logical Operation used to combine check results when both checks are active	<b>LCLMAX.CMBN</b>
Spectrum Extension Mode	Off, Range or Velocity	<b>CFARCTRL.EXTN</b>
Window Width <sup>1)</sup>	Window size to be used for local max calculation	<b>LCLMAX.WIDTH</b>
Threshold Mode	Off, Above, Under	<b>LCLMAX.TMODE</b>
Threshold Value	user defined value	<b>LCLMAX.TSHLD</b>
Threshold Extension Mode	Left or Right Justification of the TSHLD value in the 32 bit comparison value	<b>LCLMAX.LJUST</b>

1) LMODE must be set to a value of  $1_D$  or  $2_D$  and WIDTH to a value of  $1_D$  or  $2_D$  for the Local MAX Unit to flag any bins

The Unit can be configured to flag bins where the magnitude is local max or where the magnitude is not a local max. The unit can simultaneously be configured to check against a threshold value and flag a bin if the magnitude is above the threshold or if the magnitude is below the threshold.

If both checks are enabled, the results are combined as defined using the LCLMAX.CMBN bit. Both logical AND and logical OR operations can be selected. i.e. flag a bin on both checks triggering a reject or flag a bin on either check triggering a reject.

This is a 32 bit comparison. The 24 bit LCLMAX.TSHLD is extended to 32 bits before being used. This extension can either be right or left justified. If the LCLMAX.LJUST bitfield is set to  $0_B$ , the extension is right justified and 8 MSBs of value  $0_B$  are added. If LCLMAX.LJUST is set to  $1_B$ , the extension is left justified and 8 LSBs of value  $0_B$  are added.

**Note:** The Spectrum Extension function ([Section 19.3.5.5.9](#)) is used to provide the extra window data needed for the Local Max Unit window if LMODE is not equal to  $00_B$ . CFARCTRL.SEWIN must be set to a value matching the LCLMAX.WIDTH value. If LCLMAX.WIDTH= $01_B$ , then SEWIN= $000001_B$  and if LCLMX.WIDTH= $10_B$ , then SEWIN= $000010_B$ .

### 19.3.5.3 FFT Data Output Path

This is the output path for the FFT results.

The base address for the FFT results data written to Radar Memory is stored in the ODP\_CONF.BASE bitfield. The format of the data is controlled by the ODP\_CONF.MODE, ODP\_CONF.FTR, ODP\_CONF.SCALE, ODP\_CONF.EXPNT and ODP\_CONF FORMAT bitfields.

## Signal Processing Unit (SPU)

**Table 614 Output Data Processor Format Options**

Parameter	Definition	Comments
Enable FFT results output	On/Off switch to enable writing of the post-processed FFT results to memory	<b>ODP_CONF.MODE</b>
Results Compression	Enable Compression of the FFT results to 16 bit precision with a common exponent	<b>ODP_CONF.SCALE</b>
Results compression scaling factor	The scaling factor to be used if results compression is enabled. The 32 bit data will be shifted right by this number of bits before truncation to the 16 LSBs	<b>ODP_CONF.EXPNT</b>
Real Results	Force the complex component of the FFT results to 0 as it is read from the buffer memory. The complex component is written to memory but has a value of 0.	<b>ODP_CONF.FTR</b>
In Place Writing of Results	Configure the Output Data Processor to Compute Write addresses in the same order as the Input DMA so that results are written to the addresses cleared by reading input data	<b>ODP_CONF.IPF</b>
Write results as real only data	The FFT results written to memory do not have an imaginary component but consist only of packed, real values	<b>ODP_CONF.ROF</b>
Post-processing option for the FFT results	The FFT results can be converted to half precision floating point format (as defined in IEEE 754) before being written to memory.	<b>ODP_CONF.HPFP</b>

Operation of the HPFP bitfield is described in [Section 19.3.5.3.4, “Half Precision Floating Point Format” on Page 65](#).

### 19.3.5.3.1 Scalar Addition

The pre-processing unit can add a complex offset to each data value. This value is a constant stored in the SCALARADD.OPERAND bitfield and the function will be activated if a non-zero value is written to this field.

### 19.3.5.3.2 Complex Rescaling

The pre-processing unit can multiply the data value by a scaling factor. This value is a constant stored in the SCALARMULT.OPERAND bitfield which allows the value to be rescaled but does not affect the phase. The function will be activated when a non-zero value is written to this field.

### 19.3.5.3.3 Bin Rejection Unit

This units post-processes the data to be written to Radar Memory using the contents of the bin rejection register and the output from either the CFAR unit or the Local Maximum Unit.<sup>1)</sup>

The bin rejection register contains one bit for each “bin” (datapoint) in the FFT results. It is accessed through 64, 32 bit registers in the register space of the SPU. If the bit is set to 0, then the relevant bin will either be removed from the output dataset written to the Radar Memory or set to a value of  $0_B$ .

This feature can be used in conjunction with either the CFAR module or Local Maximum Unit output although here support is limited to a maximum FFT size of 1024 bits. This linking of functionality is controlled by setting the CFARCTRL.CFAREN bitfield to one of the inline operating modes. The options are Inline CFAR or Inline Local Maximum.

1) The CFAR unit will flag bins that are to be kept. The Local Max Unit will flag bins if they are to be rejected or set to  $0_B$ .

## Signal Processing Unit (SPU)

In this case, the bin rejection register and the results from the CFAR or Local Maximum Unit must both permit the bin to be written. If either function requires the bin to be removed (or set to zero), then it will be removed (or set to zero)

e.g. If the bin rejection function requires the bin to be set to  $0_D$ , then it will be set to  $0_D$  even if the CFAR has identified it as a bin to be kept.

This can be viewed as the logical ANDing of an bin acceptance mask from the CFAR/Local Max unit with the acceptance mask programmed in the Bin Rejection register. Bins corresponding to bits set to  $1_B$  in the resulting, combined mask will be retained unchanged while bins corresponding to bits set to  $0_B$  in the resulting mask will be either removed from the output or set to a value of  $0_D$  depending on the mode of the bin rejection unit

**Table 615 Bin Rejection Unit Configuration Options**

Signal	Definition	Comments
operating mode	$00_B$ = Off $01_B$ = BIN rejection, remove bins from output $10_B$ = Set BIN value to 0 $11_B$ = Reserved	<b>BINREJCTRL.RMODE</b>
BIN list to be removed	2048 bit register that defined which BIN is to be removed or zeroed (1 means BIN to be retained or passed unchanged)	<b>BINx_REJ.B</b> <b>x=0-63</b>

## Threshold Detection

The threshold detection function is integrated into the Bin Rejection Unit. The function checks to see if the bin being checked exceeds a programmed threshold. If the threshold is exceeded, then the bin is set to zero. The threshold comparison is based on the magnitude of the bin.

This is a 32 bit comparison. The 24 bit BINREJCTRL.VALUE is extended to 32 bits before being used. This extension can either be right or left justified. If the BINREJCTRL.LJUST bitfield is set to  $0_B$ , the extension is right justified and 8 MSBs of value  $0_B$  are added. If BINREJCTRL.LJUST is set to  $1_B$ , the extension is left justified and 8 LSBs of value  $0_B$  are added.

**Table 616 Thresholding Configuration Parameters**

Parameter	Definition	Comments
thresholding enable	enable / disable. If enabled, set a bin to zero if it exceeds the magnitude defined in BINREJCTRL.VALUE	Function is Combined into the Bin Rejection Unit <b>BINREJCTRL.ZMODE</b>
threshold value	user defined value	<b>BINREJCTRL.VALUE</b>
Threshold Extension Mode	Left or Right Justification of the VALUE value in the 32 bit comparison value	<b>BINREJCTRL.LJUST</b>

## Status Information

Registers are available which provide information on the behaviour of the bin rejection unit for the last FFT processed. This information is only guaranteed to be valid while the SPU is idle (i.e. at the end of each processing operation) as the data is not buffered. When the SPU is busy the information may be in the process of being updated when read. If it is desired to read the data between linked list operations, then the linked list should be configured to stop and interrupt the processor. Once the processor has read the data, SPU operation can be restarted by a write to the CTRL register.

## Signal Processing Unit (SPU)

**Note:** Any changes in the static mask set by the BINx\_BINREJ register are not buffered but are reflected immediately in the values read from the MASK\_ACCEPT registers. This happens even when the SPU is idle. The values in the MASK\_ACCEPT registers should therefore be read before any update is made to the configuration

**Table 617 Bin Rejection Unit Status**

Parameter	Definition	Comments
Rejection Count	The number of bins set to zero or rejected based on the static mask and input from the CFAR or Local Max Units	<b>BINCOUNT.REJCNT</b>
Threshold Count	The number of bins set to zero based on the threshold function of the Bin Rejection Unit	<b>BINCOUNT.THSLDCNT</b>
Acceptance Mask	The bins set to zero or rejected based on the static mask registers and input from the CFAR or Local Max Units	<b>MASK_ACCEPT.Bn</b>

**Note:** The rejection count and threshold count are calculated separately based on the bin values at the input to the unit i.e. before any processing has been applied, e.g. a bin can increment the threshold counter even if it is then rejected.

### 19.3.5.3.4 Half Precision Floating Point Format

The FFT results can be reformatted in IEEE 754 binary16 (half precision) floating point format. This is enabled by setting the bit **BEx\_ODP\_CONF (x=0-1).HPFP** to 1<sub>B</sub>

The half precision floating point format has:

- a sign bit
- 5 bits of exponent encoding a value between -14 and 15
- 11 bits of significand precision<sup>1)</sup>

With this format, the magnitude which can be encoded ranges from a minimum of  $1 \times 2^{-14}$  (neglecting subnormal numbers<sup>2)</sup>) to a maximum of  $(2-2^{-10}) \times 2^{15}$  which is equivalent to  $2^{16}-2^5 = 65504$ .

The maximum range of a 32 bit integer is from -2147483648 (0-2<sup>31</sup>) to 2147483647 (2<sup>31</sup>-1). To best fit this into the range of the half precision floating point format, the 32 bit precision number is first divided by 2<sup>15</sup> to give a range of -65536 (0-2<sup>16</sup>) to (neglecting the fractional component<sup>3)</sup> 65535 (2<sup>16</sup>-1). This is then converted to the half precision floating point format with values less than -65504 or greater than 65504 saturating at the relevant limit.

The rounding used during this conversion is “round to nearest, ties to even”.

1) 10 bits are stored to specify the fractional part of the significand, with an implicit lead bit. The lead bit value is 1 unless the exponent field is stored with all zeros  
 2) Support for subnormal numbers is not implemented. Any conversion result which would require representation using a subnormal number is instead represented as 0.  
 3) With fractional component 65535.99997

## Signal Processing Unit (SPU)

### 19.3.5.4 Non-Coherent Integration

The non-Coherent Integration<sup>1)</sup> (NCI) Module combines the data from multiple antennae into a single output. It achieves this by combining the equivalent linear power values from each antenna stored in the Buffer RAM using a scalar multiplication of the sample point followed by an addition. Each antenna can have its own multiplication constant. The multiplication constants are a 16 bit, unsigned, scalar quantity. The summed result can then be scaled to prevent overflow. The scaling operation is configured using the NCICTRL.SCALE bitfield.

The format of the output data is controlled by the NCICTRL.FORMAT bitfield. Data can be written as 16 bit or 32 bit precision real values. Additionally, an option is reserved to allow 16 bit precision complex format to be written for later reprocessing by the bin rejection unit.

This mode expects that the Input DMA Engine has been configured in Integration Mode so that the buffer RAM contains one FFT result from each connected antenna. If the data in the buffer memory is complex, then a 32 bit linear power value is calculated. If the data in the buffer memory is already a power value, this will be used directly.

Note that, if the NCI module is enabled, the CFAR module can only be used to process the NCI output, as reading data for direct input to the CFAR and the NCI require different addressing modes for the Buffer RAM.

This operation assumes that the linear power is a 32 bit, unsigned quantity. This is multiplied by the 16 bit weighting constant to give a 48 bit result which is then added into a 51 bit accumulator. Once the power values derived from the equivalent bins of all FFT results in the buffer RAM are added into the accumulator, the accumulator result can then be right shifted by 0, 1, 2 or 3 bits depending on the NCICTRL.SCALE setting.

The 32 bit output value will be taken from bits [47:16] of the accumulator value. The operation is equivalent to dividing by  $2^{16}$  and then rounding down to the nearest integer.

If a 16 bit precision result is configured this will be derived from bits [47:32] of the accumulator. The operation is equivalent to dividing the accumulator by  $2^{32}$  and rounding to the nearest integer.

For either output format, If any of bits [50:48] of the accumulator result are set, then an overflow condition is detected and all of bits of the output will be set to saturate the output value.

**Table 618 Vector Add/Sum Configuration Parameters**

Parameter	Definition	Comments
enable / disable	enable or disable the Vector add/sum <sup>1)</sup>	<b>NCICTRL.EN</b>
operand output format	16 or 32bits	<b>NCICTRL FORMAT</b>
weights	individual weights per antenna	16bit precision <b>NCISCALARx.ANTy</b> <b>x=0-3, y=0-7</b>
number of active antenna	per SPU	derived from the configuration of the Input DMA Engine and passed via Streaming Processor 1 and the FFT accelerator
result rescaling	optional result division by 1 or by 2 or by 4 or by 8.implemented using shift BUT sign extension is maintained.	to avoid saturation when having multiple antenna <b>NCICTRL.SCALE</b>

- 1) If the integrated data will neither be used by other processing units nor written to Radar Memory then enabling this bit will have no effect on processing flow i.e. the extra read pass through the buffer memory will not take place.

1) Power Domain Vector Addition

## Signal Processing Unit (SPU)

### 19.3.5.5 Constant False Alarm Rate Module

The CFAR module is used to identify interesting points in the data and then filter the data so that only these points are written to Radar Memory. It can operate in two modes as detailed in the following sections

#### 19.3.5.5.1 Inline Mode

In inline mode, the first FFT result of the measurement cycle<sup>1)</sup> is read from RAM and used to identify which datapoints are of interest. This information is then stored and used, in combination with the bin rejection register value, to filter the first and all subsequent datasets. This mode therefore requires additional time and buffer RAM read accesses to read the first FFT data for processing by the CFAR or Local Maximum Unit. The number of extra reads and therefore the delay will be dependent on the size of the CFAR window and whether or not spectrum extension is enabled.

In-line mode cannot be used if NCI is enabled.

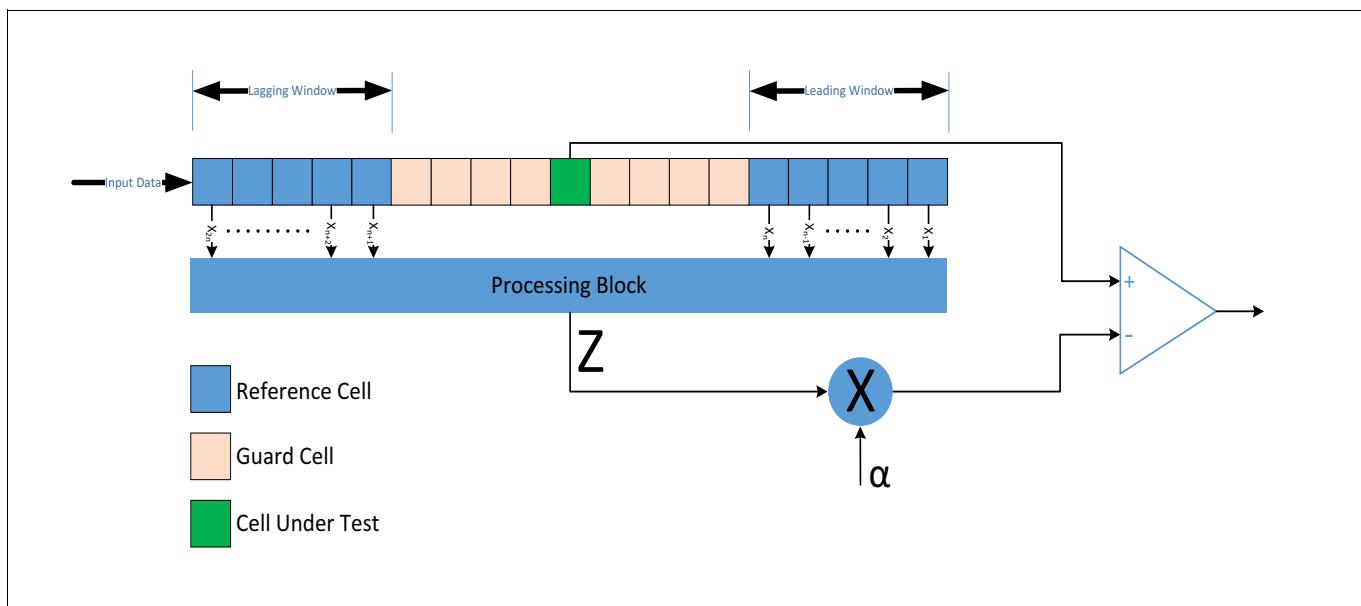
#### 19.3.5.5.2 Off-line Mode

In this mode, each ramp from each antennae is used to generate a set of data. This data is then written to Radar Memory as a table. No information is fed to the bin rejection unit.

In off-line mode, the Non-Coherent Integration Unit can be used as a pre-processor. As the data is read from the memory, it is routed through the NCI Unit to produce a composite sample value. This is then used for the CFAR input.

#### 19.3.5.5.3 CFAR Engine Architecture

The following diagram shows the top level architecture of the CFAR engine



**Figure 252 CFAR Engine Architecture**

Input power samples are shifted through the processing window, which consists of windows of reference cells on either side of the Cell Under Test (CUT). Each window is separated from the Cell Under Test by a configurable

1) If double pass mode is enabled then the process of identifying datapoint of interest is repeated for the first FFT of the measurement cycle for each pass and both sets of information are stored. The appropriate set of data will then be used for processing all further FFTs of the relevant pass.

## Signal Processing Unit (SPU)

number of guard cells. The windows cells are processed to generate a threshold value (Z). This is then scaled for comparison with the CUT value. Note that in this diagram, the threshold is shown as Z multiplied by a constant "alpha". The actual implementation outputs Z as  $\log_2$  power, not linear power so the actual implementation is  $\log_2(Z) + \text{"beta"}$ , where beta is  $\log_2(\text{alpha})$ . The parameter beta is available as a register bitfield for both engines.

### 19.3.5.4 CFAR Engine Data Format

The CFAR engine uses a 64 bit linear power value internally. This is obtained from the 32 bit values computed by the SPU logic by multiplying by  $2^{31}$  to generate a maximum possible value of  $7FFF\_FFF\_8000\_0000_H$ . Where the CFAR engine uses  $\log_2$  power for internal calculations or comparison, this is computed as a 16 bit log2 value with 7 integer and 9 fractional bits from the 64 bit linear power value.

### 19.3.5.5 CFAR Module Configuration

The CFAR module contains two, independent engines. One engine implements the CA-CFAR algorithm and some derivatives (including CASH-CFAR). The other implements the GOS-CFAR algorithm and some derivatives. Both engines can be independently enabled and configured. The exception is when the spectrum extension is enabled. In this case both engines must use an identical window size, if both are enabled.

In off-line mode, the engines are controlled by the CFARCTRL.CFAR\_CAE bitfield for the CA engine and the CFARCTRL.CFAR\_GOSE bitfield for the GOS engine. When the appropriate bitfield is set the related engine will output data to Radar Memory. Both engines will write results to Radar Memory if enabled.

The CFAR results can also be used to configure the bin rejection unit (inline mode). This is enabled by setting the CFARCTRL.CFAREN bitfield to CFARI and using the CFARCFG.CFARSEL bitfield to select an engine. If an engine is enabled, it will write its results to Radar Memory even if not selected by the CFARCTRL.CFARSEL bitfield. In in-line mode, the CFAR results are only calculated for the first FFT of the measurement cycle so a maximum of two values will be written.

#### Notes

1. *CFARCTRL.CFAREN bitfield should not be set to CFARI, CFARO or CFARN without using either CFARCTRL.CFAR\_CAE or CFARCTRL.CFAR\_GOSE to enable a CFAR engine as this will affect performance and cause the Bin Rejection Unit to behave in an unpredictable manner.*
2. *If double pass is enabled the in-line bin rejection mask will be calculated for the first FFT of both passes. The independent bin rejection masks will then be used for all further FFTs of each relevant pass*

The results data will be written to Radar Memory at incrementing contiguous addresses starting from the programmed base address. Results will be in ascending bin order with bin 0 stored at the LSB of the word at the lowest address. Results of less than 256 bits in length will be aligned on a power of 2 bit address boundary i.e.

- results of more than 8 bits size and less than 17 bits size will be aligned on 16 bit boundaries (one result in every 2 bytes of memory) results of more than 16 bits size and less than 33 bits size will be aligned on 32 bit boundaries (one result in every 4 bytes of memory)
- results of more than 32 bits size and less than 65 bits size will be aligned on 64 bit boundaries (one result in every 8 bytes of memory)
- results of more than 64 bits size and less than 128 bits size will be aligned on 128 bit boundaries (one result in every 16 bytes of memory)
- results of more than 128 bits size and less than 257 bits size will be aligned on 256 bit boundaries (one result in every 32 bytes of memory)
- Results of more than 256 bits will be rounded up in size to the next 256 bit (32 byte) address boundary.
  - e.g. a result with 734 bits of data will be fitted into 768 bits (96 bytes) of memory

## Signal Processing Unit (SPU)

If both engines are enabled, the data from the two engines can be separated with the results from the CA-CFAR written to memory starting at the base address (CFARTCTRL.BASE) and the results from the GOS-CFAR written to memory starting at the base address plus offset<sup>1)</sup> (CFARCFG3.CHAN5OFFSET). If only the GOS-CFAR engine is enabled, then the offset address can be set to 0000<sub>H</sub>.

**Table 619 1D-CFAR engines Common configuration parameters**

Parameter	Definition	Comments
Base Address	Starting address for writing CA-CFAR results to memory.	<b>CFARCTRL.BASE</b> Byte Address. must be 256 bit (32 byte) aligned
Offset Address	Word address offset added to base address to generate the starting address for writing GOS-CFAR results to memory	<b>CFARCFG3.CHAN5OFFST</b> 32 byte word Address.
number of bins	from FFT8 to FFT2048. any intermediate power of 2 value supported	FFT size is passed from the loader configuration <sup>1)</sup> <b>LDR_CONF.SIZE</b>
CFAR input selection	select input from FFT engine or from non coherent integration.	note that when input from FFT engine is used, signal power will be computed on each received BIN. <b>CFARCTRL.CFAREN</b>
Spectrum extension mode	2 extension modes depending if CFAR is run on range or on dopplermode1 = spectrum extension for rangemode2 = spectrum extension for doppler	<b>CFARCTRL.EXNSN</b>
Spectrum Extension Window	number of extra cells to be added on each end of the CFAR data	<b>CFARCTRL.SEWIN</b>
CFAR enable	CFAR enabling / disabling	<b>CFARCTRL.CFAREN,</b> <b>CFARCFG.CFAR_CAE</b> and <b>CFARCFG.CFAR_GOSE</b>

- 1) If the FFT accelerator is bypassed, the size of the input data is derived from the bin loop repeat value (ID\_RM\_BLR.BLR) if data is sourced from Radar Memory or sample count (ID\_CONF.SMPLCNT) if the data is sourced from the RIF

### 19.3.5.5.6 GOS-CFAR Engine

Configuration Parameters for the GOS-CFAR Engine are detailed in the following table

**Table 620 GOS-CFAR engine SW configuration parameters**

Parameter	Definition	Comments
GOS-CFAR algorithm	selects between GO	<b>CFARCFG.GOSALGO</b>
number of cells in leading and lagging windows for GOS-CFAR	can be set to any value from 1 to 32 cells either side of the cell under test	leading and lagging window <b>CFARCFG2.GOSWINCELL</b>

1) 2048 off, 2048 bin CFAR results will occupy 512 KiB of memory. A 16 bit offset field with byte addressing allows for 128 KiB. For this reason, the offset is specified as a 32 byte aligned, word address.

## Signal Processing Unit (SPU)

**Table 620 GOS-CFAR engine SW configuration parameters (cont'd)**

Parameter	Definition	Comments
number of guard cells	can be set to any value from 0 to 31	any value from 0 to 30 permitted <b>CFARCFG.GOSGUARD</b>
GOS-CFAR Beta parameter	16 bit, unsigned number with 7 integer bits and 9 fractional bits. Used to adjust the threshold i.e. Cell Under Test must exceed the trigger value by more than this parameter. The parameter represents $\log_2$ power	<b>CFARCFG3.GOSBETA</b>
GOS-CFAR Lead Index	Index of sorted statistic in leading window in GOS-CFAR	<b>CFARCFG2.IDXLD</b> min value 0, max value GOSWINCELL-1
GOS-CFAR Trailing Index	Index of sorted statistic in lagging window in GOS-CFAR	<b>CFARCFG2.IDXLG</b> min value 0, max value GOSWINCELL-1

In the GOS-CFAR algorithms, the samples within the leading/lagging windows are processed through a non-linear sorting operation and selected ordered statistics:  $Y_1$ ,  $Y_2$  are extracted from the two sub-windows, respectively. The GOSCA, GOSGO, and GOSSO algorithms differ in the manner  $Y_1$ ,  $Y_2$  are combined:

- GOSCA:  $Z = Y_1 + Y_2$
- GOSGO:  $Z = \max\{Y_1, Y_2\}$
- GOSSO:  $Z = \min\{Y_1, Y_2\}$

The resulting statistic  $Z$  is finally scaled by the “beta” scaling factor in order to determine the threshold value. The value of the “beta” parameter determines the probabilities of False Alarm (FA) and Missed Detection (MD). For the GOSGO and GOSSO algorithms the  $\log_2$ -domain input power sample implementation is directly equivalent to the linear one, since both the sorting and max/min operations produce equivalent ( $\log_2$ -domain) values for  $Z$ . The only difference is that the threshold value is produced by scaling  $Z$  additively with “beta”. On the other hand a direct  $\log_2$ -domain implementation of the GOSCA algorithm is not normally equivalent to the linear-domain implementation, since  $\log_2(Y_1 + Y_2)$  is not equal to  $(\log_2(Y_1) + \log_2(Y_2))$ . However in the CFAR module an equivalent  $\log_2$ -domain result to the linear domain one is obtained by post processing.

### 19.3.5.5.7 CA-CFAR Engine

The CA-CFAR engine supports the CA-CFAR, CAGO-CFAR, CASO-CFAR and CASH-CFAR algorithms. The configuration Parameters for the CA-CFAR Engine are detailed in the following table:

**Table 621 CA-CFAR engine configuration parameters**

Parameter	Definition	Comments
CA-CFAR algorithm	selects between CA, CAGO,CASO and CASH	<b>CFARCFG.CAALGO</b>
CA-CFAR Beta parameter	16 bit, unsigned number with 7 integer bits and 9 fractional bits. Used to adjust the threshold i.e. Cell Under Test must exceed the trigger value by more than this parameter. The parameter represents $\log_2$ power	<b>CFARCFG.CABETA</b>
number of guard cells	can be set to any value from 0 to 16 <sup>1)</sup>	any value from 0 to 16 permitted <b>CFARCFG.CAGUARD</b>

## Signal Processing Unit (SPU)

**Table 621 CA-CFAR engine configuration parameters (cont'd)**

Parameter	Definition	Comments
number of cells in leading and lagging windows for CA-CFAR	This defines the window size for comparison. The window size used is $2^n$ , where n is the value in the register bitfield	leading and lagging window <b>CFARCFG.CAWINCELL</b> min value 1, max value 5
active cells in window to be used for averaging for CA-CFAR/CASH-CFAR	This defines the window size for averaging within the leading and lagging windows. The window size used is $2^n$ , where n is the value in the register bitfield	<b>CFARCFG2.CASHWIN</b> min value 0, max value 5. Must be less than or equal to CAWINCELL value when CASH algorithm is selected. Otherwise must be equal to CAWINCELL

1) Values greater than 16 will cause undefined operation and incorrect results

In the CASH and CA-CFAR algorithms, the samples within the leading/lagging windows are processed through sample averaging to produce  $Y_1$ ,  $Y_2$  statistics for the leading and lagging windows respectively. Averaging is performed in the linear power domain and conversion into the  $\log_2$ -domain is performed prior to the application of the non-linear/linear operation in the processing block as shown in [Figure 252 “CFAR Engine Architecture” on Page 67](#).

In CASH-CFAR, the  $\log_2$ -converted averaged power samples are processed through MAX and MIN operations, as described in “F. X. Hofele, An Innovative CFAR Algorithm, CIE Intern. Conf. on Radar, 2001” and associated patents.

In CA/CAGO/CASO-CFAR the  $\log_2$ -domain processing is similar to the GOS-CFAR.

Threshold scaling in CASH/CA-CFAR is performed additively using the “beta” parameter in a similar manner to that used by GOS-CFAR.

### 19.3.5.8 CFAR Engine Configuration Restrictions

The CFAR engine requires some constraints on configuration to avoid undefined results

- The value of two times the sum of the number of guard cells and the number of cell in the window must not exceed the FFT size minus one.
- If spectrum extension is enabled, the spectrum extension window must be greater than or equal to the sum of the number of guard cells and the number of window cells. If both CFAR engines are enabled, this number must be calculated for both engines and the larger value must be used.
- The achievable maximum value for the  $\log_2$  power representation of the linear power is  $7E00_H$  (for a linear power of  $FFFF\_FFFF\_FFFF\_FFFF_H$ ) If the Beta parameter exceeds this value, it is impossible for any detection to occur.

### 19.3.5.9 CFAR spectrum extension

2 extension modes need to be considered: range extension and velocity extension.

Range and Velocity extension is supported by shifting additional data points into the CFAR unit at the beginning and end of the FFT to preload the window cells with the correct data. This requires the equivalent number of output data points from the CFAR to be ignored. This will be handled transparently by the supporting logic if this mode is enabled

Spectrum extension is always done per FFT dataset by adding additional bins to the dataset. The content replicating the value of some of the existing bins into the new bins. Data is fed into the CFAR unit on a “per antenna” basis. Spectrum Extension never uses data from more than one antenna. The bins to be replicated depend on the spectrum extension mode.

---

## Signal Processing Unit (SPU)

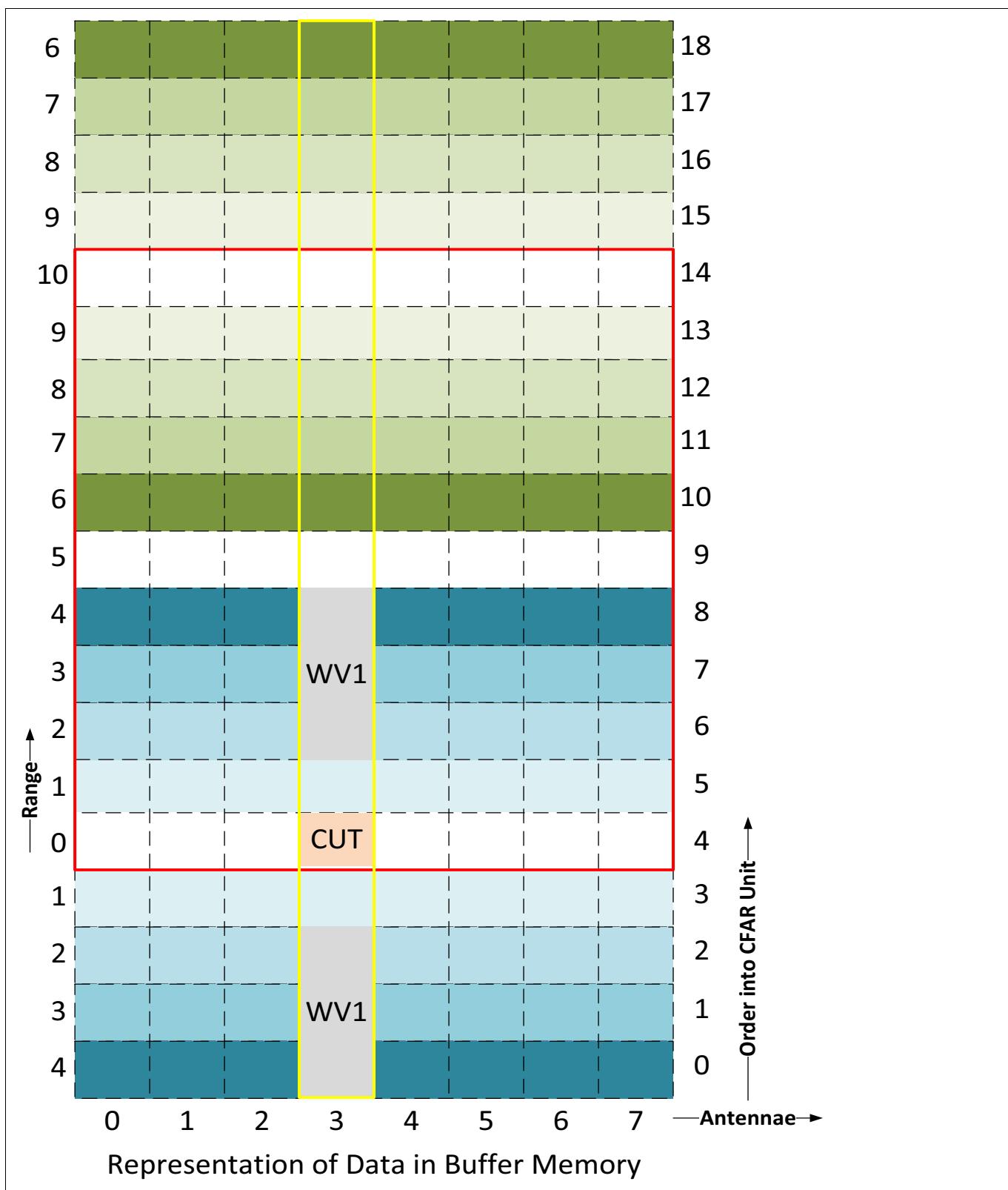
Spectrum extension will cause some data to be read twice from the buffer memory. The amount of extra reads will be directly linked to the size of the extension defined by CFARCTRL.SEWIN

**Note:** *Spectrum Extension is used in conjunction with the CFAR or Local MAX functions and should not be enabled if neither of these functions is enabled. Enabling the Spectrum Extension in the absence of enabling either CFAR or Local Max is not supported and will result in abnormal operation of the SPU.*

### Spectrum Extension for CFAR in Range

In range spectrum extension, the spectrum is extended by mirroring across the border as illustrated in the figure below. In this figure, the y-axis is along the FFT dataset (bin number ascending from the bottom of the figure).

### Signal Processing Unit (SPU)



**Figure 253 Diagram of Range Spectrum Extension**

Please note that the figure is only showing the principle as real extension needs to be done according to the window size for range and for velocity directions. The extent of the “real” data is shown by the red rectangle. The effective size of the dataset and the mirroring used to achieve it is shown by the colour coding. The extended

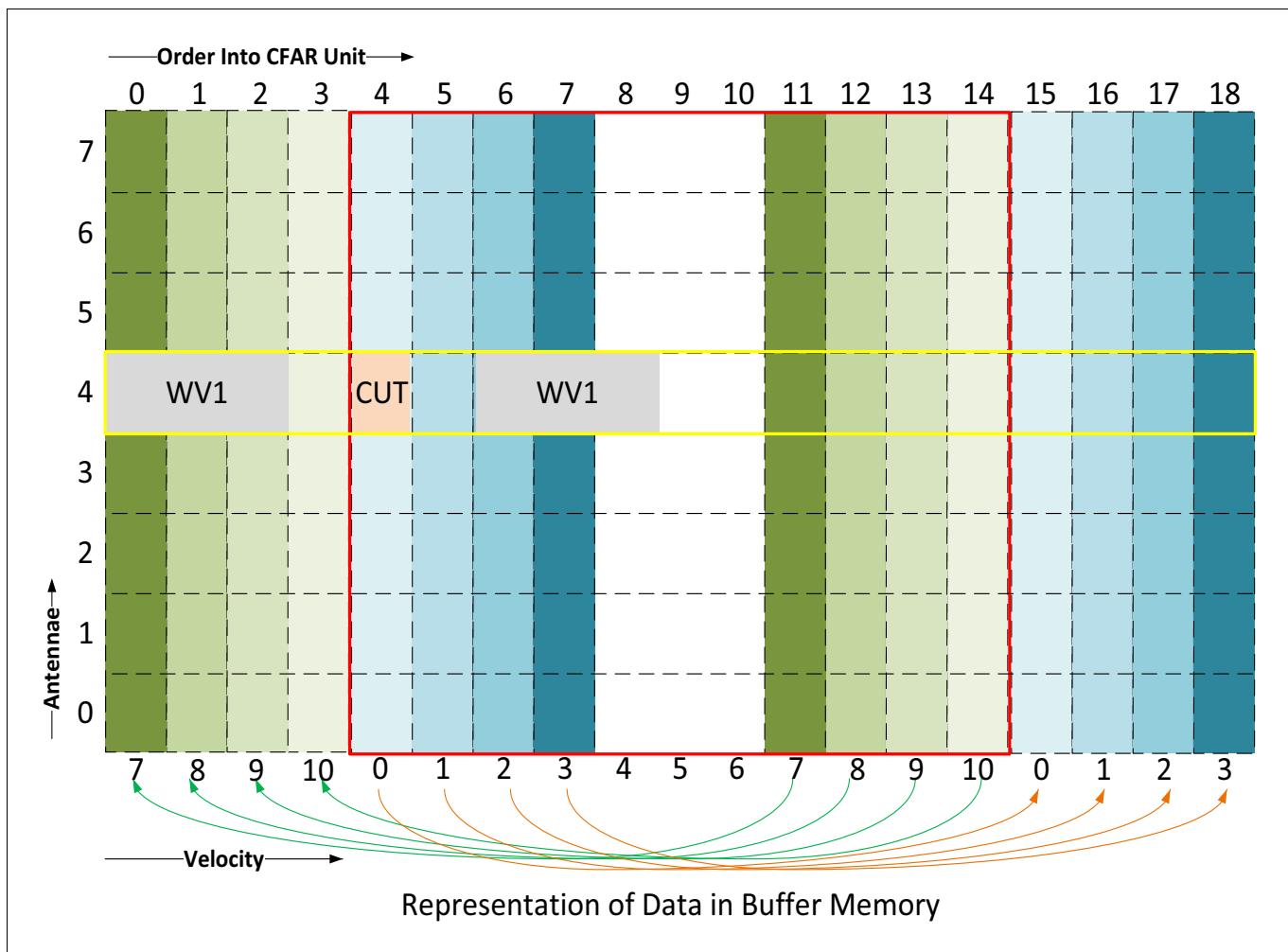
## Signal Processing Unit (SPU)

dataset fed to the CFAR units is shown as the yellow rectangle. CUT is “Cell Under Test” and WV1 represents an example window with guard band.

The behaviour if the window size exceeds the size of the dataset is not defined.

### Spectrum Extension for CFAR in Velocity

In velocity / Doppler direction, the spectrum is extended by implementing cyclic repetition. The colour coding in the diagram below shows how the repetition is implemented. The extent of the “real” data is shown by the red rectangle. Please note that the figure is only showing the principle as real extension needs to be done according to the window size for range and for velocity directions.



**Figure 254 Diagram of Velocity Spectrum Extension**

The extent of the “real” data is shown by the red rectangle. The effective size of the dataset and the mirroring used to achieve it is shown by the colour coding. The extended dataset fed to the CFAR units is shown as the yellow rectangle. CUT is “Cell Under Test” and WV1 represents an example window with guard band.

The behaviour if the window size exceeds the size of the dataset is not defined.

## Signal Processing Unit (SPU)

### Spectrum Extension on NCI Unit Output

Spectrum extension is always a one-dimensional operation which affects the input to the CFAR unit. When using spectrum extension with NCI, the integration is performed first and the spectrum extension is then applied to the resulting one-dimensional vector as if dealing with a single antenna system.

#### 19.3.5.5.10 Operation of Spectrum Extension

The spectrum extension modes work by adding extrapolated bins to each end of the FFT results. The number of bins added depend on the window size in use. For a window of e.g. 7 bins on each side of the bin under evaluation and additional 7 extrapolated bins will be added to each end of the FFT data to ensure that the first and last real bins are evaluated against a full set of window data.

Operation of the spectrum extension modes is controlled by the bitfield CFARCTRL.EXTNSN

For range spectrum extension, the extrapolated bins are mirrored. This means that when bin 0 is under comparison, real bins 1 to n are used for the values of extrapolated bins -1 to -n respectively (where n is determined by the comparison window size). For a 256 point FFT real bins 254 to 254-n are used for the values of extrapolated bins 256 to 256+n when evaluating bin 255.

For velocity spectrum extension, the extrapolated bins are determined by cyclic repetition. Again, using a 256 bin FFT as an example, when evaluating bin 0, extrapolated bins -n to -1 will be the values of real bins 255-n to 255. When evaluating bin 255, extrapolated bins 256 to 256+n will be the values of real bins 0 to n.

#### 19.3.5.6 Summation Sideband Operations

The Output Data Processor can generate several types of sideband data related to the FFT results being processed. These operations can either use log<sub>2</sub> power, linear power or approximate magnitude and can be subdivided into two types:

- Complex Arithmetic Operations
  - SUMCTRL.SUMMODE=SUMANT. Calculate the complex sum across FFT results in buffer memory (sum-antenna). The result is a 1D vector, with the same length as the FFT. Data format is defined by PWRSUM.PRECISION and can either be 16 bit precision complex or 32 bit precision complex. The result will be normalised as configured by PWRSUM.SCALE.
  - FFT. SUMCTRL.SUMMODE=SUM. Calculate the complex sum of all datapoints of an FFT. Data in this mode is always 32 bit precision. One value is stored in the output data for each FFT processed. The output value will always be normalised by dividing by the number of datapoints in the FFT<sup>1)</sup>.
  - FFT. SUMCTRL.SUMMODE=SUMLINP. Calculate the sum of the linear power of all datapoints in an FFT. Data in this mode is always 32 bit precision. One value is stored in the output data for each FFT processed. No complex part will be stored. The output value will always be normalised by dividing by the number of datapoints in the FFT<sup>1)</sup>.
- log<sub>2</sub> Power Operations
  - the sum of the log<sub>2</sub> power of all the datapoints in an FFT. SUMCTRL.PWRMODE=SUM. This will always be 16 bit precision<sup>2)</sup>. No complex part will be stored. The output value will always be normalised by dividing by the number of datapoints in the FFT<sup>1)</sup>.

Data will be written as follows

---

1) If the FFT bypass is enabled, then the number of data points in the data may not be a power of two. In this case, the dataset will be padded with zero values up to the next power of two and normalisation will divide by the power of two.  
 2) When normalising the output value to 16 bit precision, the result will be rounded down to the nearest integer. This is an exception to the general usage of “round to nearest integer”

## Signal Processing Unit (SPU)

- The output from the Complex Arithmetic Operation, if enabled, will be packed and written to ascending, contiguous addresses starting from SUMCTRL.BASE
- The output from the  $\log_2$  power operation, if enabled, will be packed and written to ascending, contiguous addresses starting from PWRSUM.BASE

The results of the Complex Arithmetic Operations (SUMMODE=SUMANT or SUMMODE=SUM) can be forced to have a zero imaginary component by setting SUMCTRL.REAL to  $1_B$ .

**Table 622 Sum Configuration Options**

Parameter	Definition	Comments
Address	Base Address for starting the writing of results	<b>SUMCTRL.BASE</b>
sum type selection	select between "off", "sum", "sum-linp" and "Sum-antenna"	<b>SUMCTRL.SUMMODE</b>
sum type selection (power data)	select between "off" and "sum" (on)	<b>SUMCTRL.PWRMODE</b>
real or complex data	select which part of the data to take as input	when real is selected, even is complex data is provided, only the real part will be taken for the sum <b>SUMCTRL.REAL</b>
Antennae	which antennae to include in Sum-antenna mode	Bitmask containing one bit per antenna <b>SUMCTRL.USEANT</b>
Address for $\log_2$ power	Base Address for starting the writing of results	<b>PWRSUM.BASE</b>

### 19.3.5.6.1 Organisation

The implementation of this function is split into three separate units in the MATH2 unit. One of these units operates on  $\log_2$  power information, one on complex data or linear power and the final unit performs the complex addition of FFT results across antenna. This is shown on the MATH2 unit block diagram. These units are

- Summation Unit (Power Sum)
- Summation Unit (Complex Sum)
- Complex Vector Add (sum-antenna or Coherent Integration Output)

### 19.3.5.7 Statistical Information

The output of Statistical data is enabled by setting SBCTRL.EN to  $1_B$ .

The mean value computed by the Unloader Unit is queued for writing to the Radar Memory. Additionally, the mean value is used to compute the variance (standard deviation squared) for each FFT. While the FFT data is being parsed to compute the variance, the minimum and maximum power values will be captured

Along with the minimum and maximum power values, the index of the FFT bin at which the minimum and maximum occurred will also be captured.

In the event of multiple bins having the same minimum or maximum, only the index of the first bin to have the value will be captured.

The base address for the writes to start is defined in SBCTRL.BASE. and configuring either of the enable conditions will result in the statistical information output channel being enabled. However, only the requested data will be written. When SBCTRL.EN is 1 then the statistical information will be written in packets of 32 bytes

## Signal Processing Unit (SPU)

per FFT (256 bit aligned) with 20 bytes of data and 12 spare bytes of indeterminate value. See [Table 623 “Statistical Information Written” on Page 77](#) for the data format used.

**Table 623 Statistical Information Written**

Parameter	Size (bytes)	Address Offset
Minimum Power	4 <sub>D</sub>	0 <sub>D</sub>
Maximum Power	4 <sub>D</sub>	4 <sub>D</sub>
Mean Power	4 <sub>D</sub>	8 <sub>D</sub>
Variance	4 <sub>D</sub>	12 <sub>D</sub>
FFT Bin Index for Minimum Power	2 <sub>D</sub>	16 <sub>D</sub>
FFT Bin Index for Maximum Power	2 <sub>D</sub>	18 <sub>D</sub>

The configuration parameters affecting the statistical data output are shown in [Table 624 “Statistical Unit Configuration Parameters” on Page 77](#)

**Table 624 Statistical Unit Configuration Parameters**

Parameter	Definition	Comments
statistical data enable	enable / disable	<b>SBCTRL.EN</b>
FFT Output Channel Enable	enable/disable FFT Result output	ODP_CONF.MODE
FFT Output Channel Post-Processing	Enable FFT Result Post-processing	ODP_CONF FORMAT

### 19.3.6 Output DMA Engine

This section describes the operation of the DMA engine that the SPU uses to write results to memory.

#### 19.3.6.1 Output DMA Engine Channels

As described in [Table 612 “Output DMA Engine Data Streams” on Page 58](#), the Output DMA Engine has eight independent data channels. These are configured using the following parameters:

**Table 625 Output DMA Engine Configuration Parameters**

Parameter	Definition	Comments <sup>1)</sup>
DMA base address for port1	DMA base address when starting a new FFT sequence from Radar memory	FFT results only <b>ODP_CONF.BASE</b>
DMA inner loop address offset	offset address to be added to base address	FFT results only
DMA outer loop address offset	offset address to be added to base address	These parameters are used only for in-place FFT and therefore have to match the Input Data Engine Configuration. Consequently these parameters are inherited from the Input DMA Engine
DMA inner loop repeat value	number of times of inner loop execution	
DMA outer loop repeat value	number of times of outer loop execution	
effective DMA transfer counter	dedicated counter to count effective number of Radar memory writes	FFT results only safety function

## Signal Processing Unit (SPU)

**Table 625 Output DMA Engine Configuration Parameters (cont'd)**

Parameter	Definition	Comments <sup>1)</sup>
DMA base address for port2	signal power	PWRCTRL.BASE
DMA base address for port3	min, Max, average	SBCTRL.BASE
DMA base address for port4	thresholding with local max or CFAR CA	CFARCTRL.BASE
DMA base address for port5	thresholding with CFAR GOS	CFARCTRL.BASE + CFARCFG3. CHAN5OFFST
DMA base address for port6	Vector add /sum	NCICTRL.BASE
DMA base address for port7	sum, sum-linp, sum-antenna	SUMCTRL.BASE
DMA Base Address for port8	sum of $\log_2$ power over FFT	PWRSUM.BASE
signed FFT BIN casting/rescaling	defines how many bits are kept from the LSB (of the 32bit internal precision) note that the sign, located on the MSB, is maintained.	ODP_CONF.EXPNT
output FIFO flush <sup>2)</sup>	FIFOs are flushed upon each configuration change. Required for the last computation step	This is an implicit parameter triggered internally when the processing of the last FFT in the measurement cycle is completed

1) All Registers here (except CFARCFG3) are duplicated to allow Double Pass Mode not to overwrite data

2) Automatic Operation, no specific control bit planned for this

Each of the BASE fields defines the start address for the output channel. The write address will be incremented automatically so that the results of each channel are stored as an array in memory starting from the defined base address.

The exception is the “In Place FFT” mode for writing the FFT results. Here the ODP\_CONF.BASE field is used as the start address but the address sequence is calculated using the Input DMA Engine Loop Count and Loop Offset parameters.

The output data stream is pushed to Radar Memory without any possibility of delay or wait states. The Radar Memory is required to keep pace with the output data rate. This is achieved by giving writes absolute priority in arbitration and arranging separate arbitration for each 256 KiB Radar Memory tile.

**Note:** *In systems with two active SPUs or with ID\_CONF2.BYPASS enabled, it is mandatory to ensure that the output from each SPU is directed to different tiles of the Radar Memory. In the event of arbitration being required between two SPU writes, then the write which loses arbitration will be lost.*

**Note:** *The SPU is specified to cope with Radar Memory sizes up to 16 MiB. Any MSBs of address bitfields not needed to address the Radar Memory available in the product will be silently ignored and set to 0<sub>B</sub> by the Output DMA Engine when initialising the output addresses. If incrementing addresses during operation causes the Output DMA Engine to generate an address outside the range of the available Radar Memory in the product, an error will be generated.*

### 19.3.6.2 Data Cube Organisation and Size after processing ADC data

The FFT results of processing ADC sample sets (ramp data) are always kept together in memory.

## Signal Processing Unit (SPU)

Each FFT result always starts at a 256 bit (32 byte) aligned address. This is mandatory if the data is to be read back into the SPU as the read operations controlled by the Input Data Manager require the start address of each dataset to be 256 bit aligned.

The ramp data are then grouped by antennae so the ramp  $n^1)$  data for all antennae (inner loop output) is grouped together in memory. The size of the data cube is variable and depends on the following factors

- data format, real or complex
- data precision 16 bit or 32 bit (DPR)
- samples per ramp or FFT output (maximum bin loop count, BLC)
- number of antennae (NA)
- number of ramps in measurement cycle (maximum outer loop count, OLC)

The size of each data point (SDP) is as follows

- 16 bit real: 2 bytes (DPR = 16 bit)
- 32 bit real: 4 bytes (DPR = 32bit)
- 16 bit complex: 4 bytes (DPR = 16 bit)
- 32 bit complex: 8 bytes (DPR = 32bit)

The size in bytes of the data generated for each configured antennae per ramp (ADS) is

(19.2)

$$\text{ADS(bytes)} = \text{BLC} \times \text{SDP}$$

The value for ADS will always be rounded up to the next 32 bytes (ADSR).

The overall dataset size (DS) is then

(19.3)

$$\text{DS(bytes)} = (\text{ADSR}) \times \text{NA} \times \text{OLC}$$

As the maximum amount of data that can fit into the Buffer RAM is 32 kBytes, this gives a hard limit for the amount of data per inner loop (the ADS value).

### 19.3.7 Radar sequencer

The Radar sequencer is the main control sequencer of the SPU. It manages the configuration/reconfiguration of each unit and controls the sequencing of operations. It also monitors each unit for correct operation and generates suitable signalling to the application software via the Interrupt system and the Safety Management Unit.

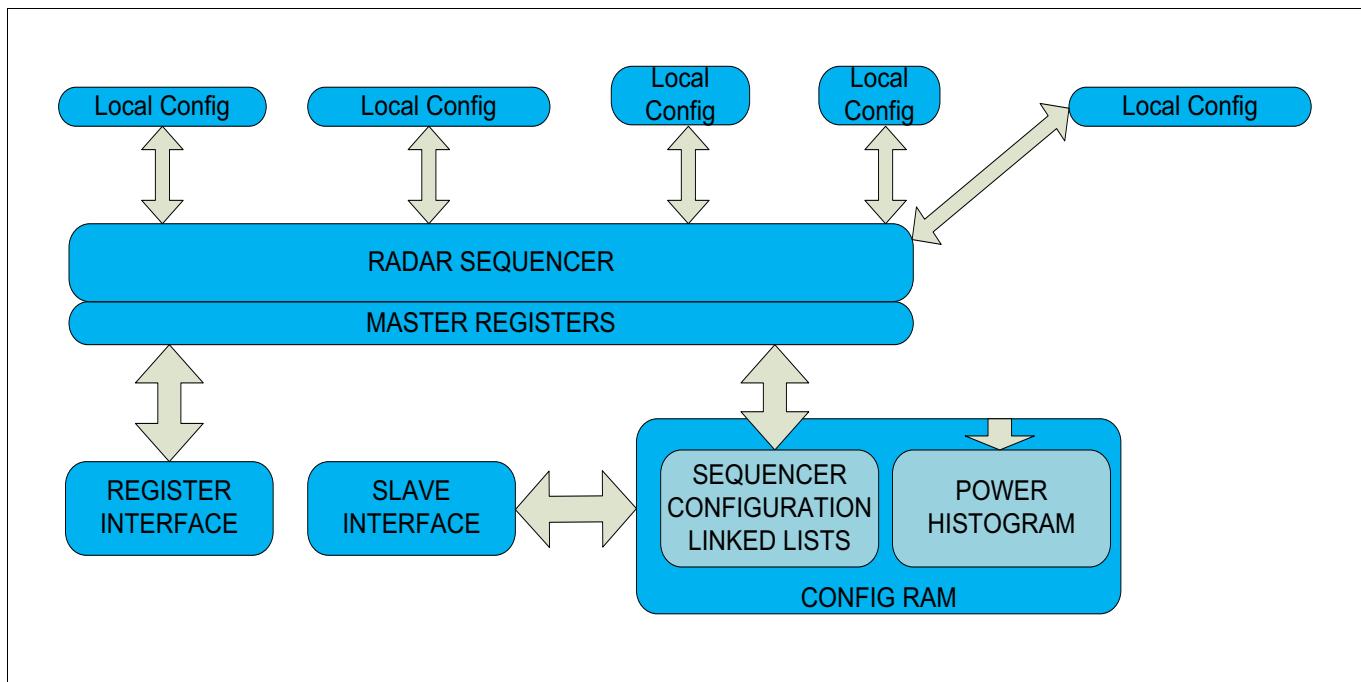
General principles of operation are:

- Configurations can be written directly to the registers of the SPU by application software
- However, configurations can also be stored in the configuration memory, which is a RAM accessible by the application software (this RAM is also used to store the histogram)
- Configurations are organized in linked lists.

This following sections describe the features and behaviour controlled by the Radar Sequencer.

1) where n is the ramp number and also the outer loop count

## Signal Processing Unit (SPU)



**Figure 255 Radar Sequencer Overview**

### 19.3.7.1 General Configuration

The Radar sequencer has several configuration options that define the overall configuration of the SPU.

**Table 626 General Configuration Options**

Function	Register	Bitfield	Comment
Clocking Mode	CTRL	DIV	FFT Operating Mode. Configures the rate at which the LOADER outputs data. 1, 2, 4 or 8 clocks/bin are possible. 2 clks/bin is considered “normal” operating speed
	CLC	DISR	Clock Disable. Switch off the SPU’s internal clock
Linked List Base Address	CTRL	NXT_CONF	Base address for the next set of configuration information in the linked list
Start	CTRL	TRIG	Starts the Radar Sequencer
Operating Mode	CTRL	MODE	Defines when the SPU will start processing input data
Attention	CTRL	ATTN	Trigger a service request interrupt when execution of the current configuration is completed
Cross Trigger	CTRL	XTRIG	Trigger a “DONE” event when execution of the current configuration is completed.
Last Configuration	CTRL	LAST	Current Configuration is the last configuration to be executed. Do not load a further configuration from the address defined by NXT_CONF

## Signal Processing Unit (SPU)

### 19.3.7.2 Radar sequencer start / stop

CTRL.MODE is used to define the conditions under which the SPU will start processing data.

The register space is organised so that the CTRL register should be the last register read during configuration by linked list. This ensures that a software trigger always starts with a valid data set. Starting processing of a linked list can be started either by writing a full set of register values directly into the register site finishing with a valid write to the CTRL register or by writing a MODE value of “RELOAD” and a valid NXT\_CONT value to the CTRL register

Writing a 1<sub>B</sub> to the CTRL.TRIG bit starts the radar sequencer provided that the CTRL.MODE bitfield is set to “SW” (software).

The SPU will stop processing a linked list when processing completes on a configuration which has the CTRL.LAST bit set or if the configuration just loaded contains “OFF” or “STOP” in the CTRL.MODE field.

**Note:** *The only difference between setting CTRL.LAST and loading a configuration set with CTRL.MODE set to OFF is that using CTRL.LAST eliminates an unnecessary configuration load. CTRL.MODE=STOP should only be used if it is required to abort processing off a linked list. If it is loaded from configuration memory then it will behave in the same way as setting CTRL.MODE to OFF.*

The CTRL.BUSY bitfield shows if the SPU is running or is stopped. Loading a configuration which has a CTRL.MODE field not set to OFF or STOP will cause the SPU to transition to the running state even if it is then just waiting for a trigger event. Once in the running state, software can only stop the SPU by writing “STOP” to CTRL.MODE. Any other value will have no effect.

**Note:** *Only the CTRL register should be written while the SPU is busy. Writing any other register will have immediate effects on the SPU configuration and may lead to undesirable consequences including a lock up of the SPU.*

**Table 627 CTRL.MODE Value Definitions**

Value	Function
OFF	SPU will do nothing. This is used to terminate a linked list operation when loaded from a configuration set in config RAM. Software writing OFF to the CTRL register once the SPU has been configured has no effect.
INT	SPU will pause waiting for data on the RIF input. Valid data on the RIF input will trigger operation. This can be qualified by the Partial Acquisition counter LIMIT value if the PACTR.TRIG bit is set
EXT	SPU will pause waiting for a trigger from an external source. This is used to trigger the SPU from the SPU Lockstep module if present on the product. Valid data on the RIF input will be ignored until the trigger event occurs. Valid data on the RIF data after the configured measurement cycle has completed will be ignored. If the Radar memory is configured as the data source, processing will commence immediately.
SPU0	SPU will trigger on an “DONE” event from SPU0 <sup>1)</sup>
SPU1	SPU will trigger on an “DONE” event from SPU1 <sup>1)</sup>
RELOAD	SPU will trigger an immediate configuration load from the value in the CTRL.NXTCONF field

## Signal Processing Unit (SPU)

**Table 627 CTRL.MODE Value Definitions (cont'd)**

Value	Function
SW	SPU will immediately start processing when a $1_B$ is detected in the CTRL.TRIG bitfield. The trigger event may occur simultaneously with "SW" being written to CTRL.MODE
STOP	This value should be written to the MODE bitfield if it is required to stop the SPU in the middle of linked list processing. If written while the SPU is processing data, the SPU will stop when the current processing operation is completed rather than loading the next configuration set. This value should not be loaded from a configuration set in RAM. In this case OFF should be used instead

1) Note that the SPU will only generate the "DONE" event when explicitly configured using the CTRL.XTRIG bitfield

The values INT and EXT are intended to be used when data is being read from one or more RIF instances (or, in the case of EXT, if the SPUs are lockstepped).

The value EXT can also be used when reading data from Radar Memory when the SPUs are lockstepped.

SPU0, SPU1 and SW are intended to be used when reading data from Radar Memory. These options cannot be used when the SPU s are lockstepped.

As soon as any of the values: INT, EXT, SPU0, SPU1 or SW are written to the CTRL.MODE bitfield, the SPU will report its state as "BUSY" when CTRL.BUSY is read.

### 19.3.7.3 Synchronized Radar sequencer Start

The synchronised start function allows the 2 SPUs to be started at the same time. When using synchronised start, there will be no skew between the start point of the two SPUs as this mode is used to start processing in lockstep mode when reading data from the Radar Memory. Synchronised start is configured setting the CTRL.MODE bitfield to EXT and using the SPU Lockstep module to simultaneously trigger both SPUs.

### 19.3.7.4 Configuration / Reconfiguration

During linked lists of computations, the Radar SPU supports full reconfiguration of the computation units of the SPU by reading the full register set from configuration memory. Partial reconfiguration is supported by making a copy of the previous configuration into the configuration memory and changing only the register fields affecting the functionality requiring reconfiguration before reloading it directly into the SPU registers using the linked list function.

### 19.3.7.5 Linked Lists Organization

A linked list is a set of configuration sets stored in Configuration Memory defining a sequence of computations that are run without CPU intervention. The linked list is stores in memory as an image of the registers, starting from IDM\_CONF and ending at CTRL. The image of IDM\_CONF is stored at the base address of the linked list entry and all the other registers are at the same relative offset as their address in the system memory map.

The Transaction Counter registers, CRC registers, CLC, MODID, STAT, MONITOR, OCS, ODA, USROTC, ACCEN, KRST0, KRST1 and KRSTCLR registers cannot be reconfigured by the linked list function.

### 19.3.7.6 CPU monitoring during run time

The CPU(s) can read configuration registers during run time to check or wait for the radar sequencer to complete a computation sequence. The CPU can also read status registers to monitor the proper execution of the sequence. The following status registers can be monitored (read only) during run time

- ramp counter (derived from Input Data Manager)

## Signal Processing Unit (SPU)

- sample counter
- 1 busy flag per sub block
- 1 busy flag for SPU
- linked list sequence completed

### 19.3.7.7 Interrupts

The Radar SPU has a high degree of autonomy. As such, it generates few interrupts. Each interrupt source can be masked but for safety reasons, its flag can be read by application software.

Each SPU instance has two physical interrupt requests.

- Attention Request: asserted on one or more of events occurring during normal operation when intervention by the application software is needed
- Error Flag: asserted on an error condition when accurate completion of the calculation is compromised

The table below is showing the different interrupt sources

**Table 628 Interrupt Trigger Definition**

Event	Definition	Comments
end of execution	A configuration with the CTRL.ATTN bit set has completed execution	Attention Request <b>STAT.INTMSK(0)</b>
Arithmetic Overflow During iFFT	The FFT accelerator has reported that an output number will not fit into the supported format	Attention Request <b>STAT.INTMSK(1)</b>
Input DMA Unit	To be defined.	Attention Request <b>STAT.INTMSK(2)</b>
end of linked list	set after confirmed write of the last result of the outer loop.	Attention Request <b>STAT.INTMSK(3)</b>
Partial Acquisition Counter Trigger	Partial Acquisition counter has reached programmed limit with PACTR.ATTN bit set	Attention Request <b>STAT.INTMSK(4)</b>
any status event (to be defined)		<b>STAT.INTMSK(5)</b>
Radar Memory Read Error	Address Overflow on Radar Memory read or Radar Memory has reported an error on read	Error Flag <b>STAT.ERRMSK(0)</b>
Input DMA Unit write error	Address Overflow on Buffer Memory write	Error Flag <b>STAT.ERRMSK(1)</b>
Radar Memory Write error	Address Overflow on Radar Memory Write or the Radar Memory has reported an error on write. This covers both the Output DMA and the Bypass Channel for RIF1 data	Error Flag <b>STAT.ERRMSK(2)</b>
FIFO overrun	One or more of the SPU internal FIFOs has overflowed	Error Flag <b>STAT.ERRMSK(3)</b>

## Signal Processing Unit (SPU)

**Table 628 Interrupt Trigger Definition (cont'd)**

Event	Definition	Comments
Partial Acquisition Counter Error	Partial Acquisition Counter has reached limit with PACTR.ERR bit set	Error Flag <b>STAT.ERRMSK(4)</b>
Input Data Overrun	The LOADER/FFT/MATH2 pipeline has not completed processing when the next set of FFTs have been fully written to buffer memory and the RIF has more data to be written for the following ramp. A ramp has (or ramps have) been skipped.	Error Flag <b>STAT.ERRMSK(5)</b>

### 19.3.8 Streaming Processor 1, Buffer RAM Switching Behaviour

When the Input DMA Engine has finished writing a data block to the Buffer RAM, it will enable the switching of the Buffer RAMs to present the new data to the Loader module. If the Loader module is still reading the previous data block, the switch will not take place until the Loader module has finished.

The behaviour then varies depending on whether the data source is the Radar Interface or Radar Memory

#### 19.3.8.1 Data Source is Radar Interface

In the event that new data is presented at the Input DMA Engine inputs before the Loader has finished, the data will be written to the RAM and the switch of the buffer RAMs will not take place. This will be flagged as an "Input Data Overrun" error and can be used to trigger an SPU error interrupt (STAT.INTMSK(5) and STAT.ERRMSK(5)). As some applications may find controlled skipping of ramps useful, the maximum number of ramps skipped between buffer RAM switches occurring in a measurement cycle will be stored in the OVRRN field of the STAT register.

#### 19.3.8.2 Data Source is Radar Memory

Reading of further data from the Radar Memory will be stalled until the Loader has finished reading the previous data and the buffer RAM can be switched.

### 19.3.9 Configuration Memory

The configuration memory is used for several functions

- Storing complete or partial sets of configuration register settings for use by the linked list function of the SPU
- Storing Window Parameters for use by the FFT.
- Storing the working data for the power histogram calculation

The Configuration Memory is mapped into the system address space so that the application software can:

- write configuration register settings.
- write window parameters.
- read power histogram information.

Note that the operation of the histogram function requires all the available bandwidth of the RAM.

If the SPU is busy, then accesses to the configuration memory will be errored to prevent the bus access stalling for an extended period.

---

## Signal Processing Unit (SPU)

### 19.3.9.1 Safety/Security

As the configuration memory is primarily used for storing configuration settings, it is protected by the Access Enable mechanism used to prevent unauthorised modification of the SPU registers.

### 19.3.9.2 Configuration Register Data Format

The configuration register settings are stored in the configuration memory in the same organisation as they exist in the register space. Registers that can be updated from the configuration memory are in the address range ID\_CONF to CTRL. The ID\_CONF value will be stored at an address of  $0_H$  relative to the start address for the configuration set. The relative addresses of the data values for the other registers remain unchanged from their address map in system memory.

### 19.3.9.2.1 Loading Configuration Settings

The loading of configuration settings into the registers can be triggered either manually or automatically by one of the following events:

- Software Trigger
- Trigger on SPU idle
  - This will trigger a complete register load when the last write is flushed to Radar Memory

### 19.3.9.3 Window Data Format

The window data is stored linearly in the configuration memory with the address offset corresponding to the data point that the window value should be used for. The address offset size is affected by the format of the window data set in LDR\_CONF2 and will vary between 2 and 8 bytes per coefficient. For complex window coefficients, the real component is stored at the lower address. An independent address offset can be defined for each antenna.

### 19.3.9.4 Configuration Memory Usage Restrictions

The configuration information is stored in a memory that is independent from the Radar memory. This memory is also used to store the results from the histogram unit and the window parameters.

The histogram unit needs all the available bandwidth of this memory. The configuration memory is therefore organised as two independent banks and, if the histogram is enabled, the window information and configuration linked lists must be stored in the other bank

The TriCore subsystem should not attempt to access this memory during SPU run time. If a processor does attempt to access the memory while the SPU is utilising it the SPU will generate an error to the TriCore subsystem. This is to avoid stalling the processor execution for an excessive period of time.

The configuration memory is organised in 8 byte words. All register fields used to store configuration memory addresses must contain 8 byte aligned values. Any non-zero value written to bits[2:0] of these register fields will be ignored.

The configuration memory does not support byte or half-word (16 bit) write accesses

**Signal Processing Unit (SPU)****19.4 Registers****Table 629 Register Overview - SPU (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Page Number</b>
CLC	Clock Control	00000 <sub>H</sub>	88
MODID	Module Identification Register	00004 <sub>H</sub>	89
STAT	Status and Reporting	00008 <sub>H</sub>	90
ID_CONF	Input DMA Configuration	00030 <sub>H</sub>	91
ID_CONF2	Input DMA Configuration 2	00034 <sub>H</sub>	93
ID_RM_CONF	Input DMA Configuration: Radar Memory	00038 <sub>H</sub>	94
ID_RM_ILO	Inner Loop Address Offset	0003C <sub>H</sub>	97
ID_RM_OLO	Outer Loop Address Offset	00040 <sub>H</sub>	98
ID_RM_BLO	Bin Offset Address Configuration	00044 <sub>H</sub>	99
ID_RM_IOLR	Inner and Outer Loop Repeat	00048 <sub>H</sub>	100
ID_RM_BLR	Bin Loop Repeat	0004C <sub>H</sub>	101
ID_RM_ACFG0	Spare Configuration Register	00050 <sub>H</sub>	102
ID_RM_ACFG1	Spare Configuration Register	00054 <sub>H</sub>	103
PACTR	Partial-Acquisition Counter	00058 <sub>H</sub>	104
DPASS_CONF	Double Pass Configuration	0005C <sub>H</sub>	105
BEx_LDR_CONF	Loader Configuration	00060 <sub>H</sub> +x*40 <sub>H</sub>	107
BEx_LDR_CONF2	Loader Configuration Extended	00064 <sub>H</sub> +x*40 <sub>H</sub>	109
BEx_Aj_ANTOFST	Antenna Offset	00068 <sub>H</sub> +x*40 <sub>H</sub> +j*4	110
BEx_UNLDR_CONF	Unloader Configuration	00078 <sub>H</sub> +x*40 <sub>H</sub>	111
BEx_UNLDR_CONF2	Unloader Configuration 2	0007C <sub>H</sub> +x*40 <sub>H</sub>	113
BEx_UNLDR_ACFG	Spare Configuration Register	00080 <sub>H</sub> +x*40 <sub>H</sub>	113
BEx_ODP_CONF	Output Data Processor Configuration	00084 <sub>H</sub> +x*40 <sub>H</sub>	114
BEx_NCICTRL	NCI Control	00088 <sub>H</sub> +x*40 <sub>H</sub>	116
BEx_SUMCTRL	Summation Unit Control	0008C <sub>H</sub> +x*40 <sub>H</sub>	118
BEx_PWRSUM	Power Summation	00090 <sub>H</sub> +x*40 <sub>H</sub>	120
BEx_PWRCTRL	Power Information Channel Control	00094 <sub>H</sub> +x*40 <sub>H</sub>	122
BEx_CFARCTRL	CFAR Module Control	00098 <sub>H</sub> +x*40 <sub>H</sub>	123
BEx_SBCTRL	Sideband Control	0009C <sub>H</sub> +x*40 <sub>H</sub>	126
BINm_REJ	Bin Rejection Mask	000E0 <sub>H</sub> +m*4	127
MAGAPPROX	Magnitude Approximation Constants	001E0 <sub>H</sub>	128
NCISCALAR0	NCI Antennae Scaling Factor	001E4 <sub>H</sub>	129
NCISCALAR1	NCI Antennae Scaling Factor	001E8 <sub>H</sub>	130
NCISCALAR2	NCI Antennae Scaling Factor	001EC <sub>H</sub>	130
NCISCALAR3	NCI Antennae Scaling Factor	001F0 <sub>H</sub>	131

## Signal Processing Unit (SPU)

**Table 629 Register Overview - SPU (ascending Offset Address) (cont'd)**

Short Name	Long Name	Offset Address	Page Number
CFARCFG	CFAR Configuration	001F4 <sub>H</sub>	<a href="#">132</a>
CFARCFG2	CFAR Configuration 2	001F8 <sub>H</sub>	<a href="#">134</a>
CFARCFG3	CFAR Configuration 3	001FC <sub>H</sub>	<a href="#">135</a>
SCALARADD	Scalar Addition Operand	00200 <sub>H</sub>	<a href="#">136</a>
SCALARMULT	Scalar Multiplication Operand	00204 <sub>H</sub>	<a href="#">137</a>
BINREJCTRL	Bin Rejection Unit Control	00208 <sub>H</sub>	<a href="#">138</a>
LCLMAX	Local Maximum Control	0020C <sub>H</sub>	<a href="#">139</a>
ACFG2	Spare Configuration Register	00210 <sub>H</sub>	<a href="#">141</a>
REGCRC	Register CRC	00218 <sub>H</sub>	<a href="#">141</a>
CTRL	SPU Control	0021C <sub>H</sub>	<a href="#">142</a>
MDq_METADATA	Dataset Metadata	00220 <sub>H</sub> +q*88 <sub>H</sub>	<a href="#">144</a>
MDq_BINCOUNT	Bin Rejection Unit Tracking	00224 <sub>H</sub> +q*88 <sub>H</sub>	<a href="#">145</a>
MDq_MASKm_ACCEP T	Bin Acceptance Mask	00228 <sub>H</sub> +q*88 <sub>H</sub> +m*4	<a href="#">146</a>
IDMCNT	Input DMA Count	00330 <sub>H</sub>	<a href="#">147</a>
IBMCNT	Input Buffer Memory Count	00334 <sub>H</sub>	<a href="#">148</a>
LDRCNT	Input Buffer Memory Read Count	00338 <sub>H</sub>	<a href="#">149</a>
FFTWCNT	FFT Load Count	0033C <sub>H</sub>	<a href="#">150</a>
FFTRCNT	FFT Unload Count	00340 <sub>H</sub>	<a href="#">151</a>
ULDRCNT	Output Buffer Memory Write Count	00344 <sub>H</sub>	<a href="#">152</a>
ODMCNT	Output Buffer Memory Read Count	00348 <sub>H</sub>	<a href="#">153</a>
BRCnt	Bin Rejection Unit Load Count	0034C <sub>H</sub>	<a href="#">154</a>
CFARCNT	CFAR Unit Load Count	00350 <sub>H</sub>	<a href="#">155</a>
ODMACNTp	Output DMA Port Write Count	00354 <sub>H</sub> +p*4	<a href="#">156</a>
CNTCLR	Safety Counter Clear	00374 <sub>H</sub>	<a href="#">158</a>
MONITOR	SPU Monitor	00378 <sub>H</sub>	<a href="#">159</a>
SMCTRL	Safety Mechanism Control Functions	0037C <sub>H</sub>	<a href="#">160</a>
SMSTAT	Safety Mechanism Status	00380 <sub>H</sub>	<a href="#">162</a>
SMUSER	Safety Mechanism Control Functions (User)	00384 <sub>H</sub>	<a href="#">164</a>
DATAd_CRC	Monitor CRC Register	00388 <sub>H</sub> +d*4	<a href="#">166</a>
CTRLe_CRC	Monitor CRC Register	00500 <sub>H</sub> +e*4	<a href="#">167</a>
USRRTC	User OCDS Trace Control	007E0 <sub>H</sub>	<a href="#">167</a>
ACCENO	Access Enable Register 0	007E4 <sub>H</sub>	<a href="#">168</a>
ACCEN1	Access Enable Register 1	007E8 <sub>H</sub>	<a href="#">169</a>
OCS	OCDS Control and Status	007EC <sub>H</sub>	<a href="#">170</a>
ODA	OCDS Debug Access Register	007F0 <sub>H</sub>	<a href="#">172</a>
KRST0	Kernel Reset Register 0	007F4 <sub>H</sub>	<a href="#">172</a>

## Signal Processing Unit (SPU)

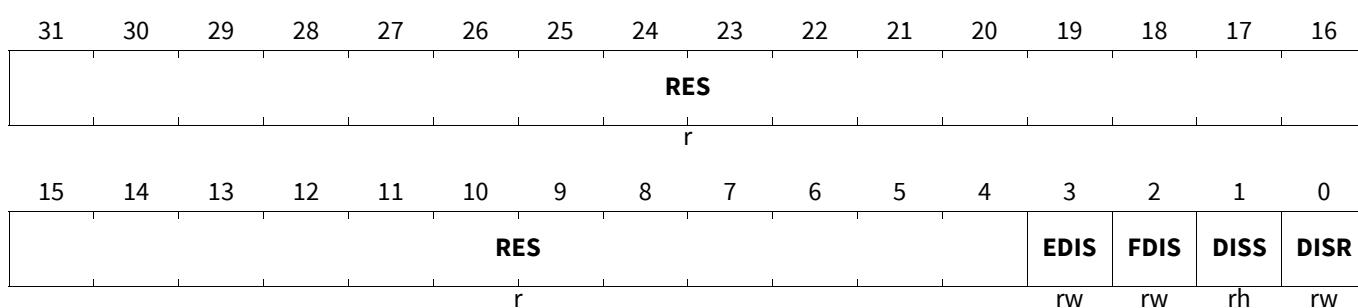
**Table 629 Register Overview - SPU (ascending Offset Address) (cont'd)**

Short Name	Long Name	Offset Address	Page Number
KRST1	Kernel Reset Register 1	007F8 <sub>H</sub>	<a href="#">174</a>
KRSTCLR	Kernel Reset Clear	007FC <sub>H</sub>	<a href="#">175</a>

### 19.4.1 Register Description

#### Clock Control

**CLC**  
**Clock Control** (00000<sub>H</sub>) Reset Value: [Table 631](#)



Field	Bits	Type	Description
<b>DISR</b>	0	rw	<b>Disable Request</b> 0 <sub>B</sub> <b>Enable</b> , Request that the SPU clock tree be switched on 1 <sub>B</sub> <b>Disable</b> , Request that the SPU clock tree be switched off
<b>DISS</b>	1	rh	<b>Disable Status</b> This bit will be set to 1 if the SPU kernel clock is disabled 0 <sub>B</sub> <b>Enable</b> , The SPU clock tree is switched on 1 <sub>B</sub> <b>Disabled</b> , The SPU clock tree is switched off
<b>FDIS</b>	2	rw	<b>Freeze Disable</b> This bit controls the freeze function for this module. The freeze function is not implemented for the SPU so this bit will have no effect. This bit can be set to 0 <sub>B</sub> 0 <sub>B</sub> <b>Disabled</b> , Module Operates on corrected clock with reduced modulation jitter. No effect for SPU 1 <sub>B</sub> <b>Enable</b> , Module operates on uncorrected clock, with full modulation jitter.
<b>EDIS</b>	3	rw	<b>Sleep Mode Enable Control</b> Reserved. This bit currently has no effect on the SPU. It should be kept at 0 <sub>B</sub> for future compatibility 0 <sub>B</sub> <b>Disabled</b> , Sleep Mode is Off 1 <sub>B</sub> <b>Enable</b> , Sleep Mode is On (no effect)
<b>RES</b>	31:4	r	<b>Reserved</b>

## Signal Processing Unit (SPU)

**Table 630 Access Mode Restrictions of CLC sorted by descending priority**

Mode Name	Access Mode		Description
ENDINIT and Master enabled in ACCEN	r	EDIS, FDIS, RES	Write Accesses Permitted during Initialisation
	rh	DISS	
	rw	DISR	
Otherwise (default)	r	DISR, EDIS, FDIS, RES	Default Access Mode (Bus Error on Write)
	rh	DISS	

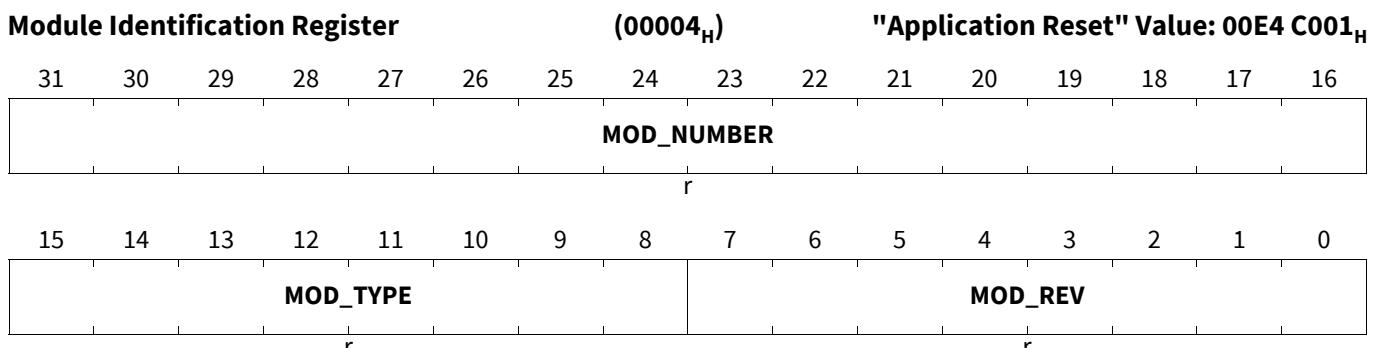
**Table 631 Reset Values of CLC**

Reset Type	Reset Value	Note
Application Reset	0000 0003 <sub>H</sub>	Application Reset

### Module Identification Register

This register contains information intended to enable identification of the module and the version of the module. The Revision Number (MOD\_REV) of the SPU is 01<sub>H</sub>.

#### MODID



Field	Bits	Type	Description
<b>MOD_REV</b>	7:0	r	<b>Module Revision Number</b> This bit field defines the module revision number. The value of a module revision starts with 01H (first revision)
<b>MOD_TYPE</b>	15:8	r	<b>Module Type</b> Set to 0xC0 to indicate a 32 bit module
<b>MOD_NUMBE R</b>	31:16	r	<b>Module Number Value</b> Set to 0x00E4. This number is unique to the SPU.

**Table 632 Access Mode Restrictions of MODID sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	MOD_NUMBER, MOD_REV, MOD_TYPE	Any write access will cause bus error
Otherwise (default)	r	MOD_NUMBER, MOD_REV, MOD_TYPE	Default Access Mode (Bus Error on Write)

## Signal Processing Unit (SPU)

### Status and Reporting

This register allows the interrupt trigger masks to be set and the triggers causing interrupts to be read. See [Section 19.3.7.7, Interrupts](#) for bit allocation in the ERRMSK, INTMSK, ERRTRG and INTTRG fields

#### STAT

<b>Status and Reporting</b>																<b>(00008<sub>H</sub>)</b>				<b>Reset Value:</b> <a href="#">Table 634</a>							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	<b>INTTRG</b>											
rh										rh										<b>RES</b>	<b>OVRRN</b>		r	rwh			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	<b>INTMSK</b>											
rw										w	rw	<b>INTCLR</b>		<b>INTSTS</b>	<b>ERRMSK</b>												

Field	Bits	Type	Description
<b>ERRSTS</b>	0	rw	<b>Error Status</b> Set when an enabled error condition has triggered an interrupt
<b>ERRCLR</b>	1	w	<b>Error Clear</b> Set this bit to 1 while writing 0 to ERRSTS to clear the ERRSTS flag. Always reads as 0
<b>ERRMSK</b>	7:2	rw	<b>Error Mask</b> Mask field to enable interrupt on particular error conditions
<b>INTSTS</b>	8	rw	<b>Interrupt Status</b> Set to 1 when an enabled interrupt condition (INTMSK) has triggered an interrupt
<b>INTCLR</b>	9	w	<b>Interrupt Clear</b> Set this bit to 1 while writing 0 to INTSTS to clear the Interrupt Status Flag. Always reads as 0
<b>INTMSK</b>	15:10	rw	<b>Interrupt Mask</b> Mask Field to Enable/Disable Service Interrupt on particular events or conditions
<b>OVRRN</b>	18:16	rwh	<b>Overrun</b> The maximum number of contiguous datasets skipped by the Input Data Manager due to overrun in the FFT processing. This field will continue to accumulate the maximum number of ramps skipped sequentially until it is explicitly cleared by writing 0. Any value written during configuration will be overwritten at this point.
<b>RES</b>	19	r	<b>Reserved</b>
<b>ERRTRG</b>	25:20	rh	<b>Interrupt Trigger</b> The bits in this bitfield correspond to the bits in ERRMSK. Bits set will indicate which condition or conditions have triggered an interrupt. This field will be cleared when ERRSTS is cleared

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
INTTRG	31:26	rh	<b>Interrupt Trigger</b> The bits in this bitfield correspond to the bits in INTMSK. Bits set will indicate which condition or conditions have triggered an interrupt. This field will be cleared when INTSTS is cleared

**Table 633 Access Mode Restrictions of STAT sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rh	ERRTRG, INTTRG	
	rw	ERRMSK, ERRSTS, INTMSK, INTSTS	
	rwh	OVRNN	
	w	ERRCLR, INTCLR	
Otherwise (default)	r	ERRMSK, ERRSTS, INTMSK, INTSTS, RES	Default Access Mode (Bus Error on Write)
	rX	ERRCLR, INTCLR	
	rh	ERRTRG, INTTRG, OVRNN	

**Table 634 Reset Values of STAT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Input DMA Configuration

This register configures the base operating mode for the Input DMA and also provides the configuration parameters specific to accepting data from the RIF(s). The ANT, SMPLCNT, RAMPS and FORMAT fields are only used if the data source in the SRC field is specified as one or more RIFs. These fields will be ignored if the SRC field specifies that data is to be read from Radar Memory

### ID\_CONF

**Input DMA Configuration (00030<sub>H</sub>) Reset Value: Table 636**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>FORM AT</b>	<b>SIGNE D</b>	<b>RES</b>			<b>RAMPS</b>										
rw	rw	r													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>RES</b>	<b>SMPLCNT</b>										<b>ANT</b>	<b>SRC</b>			
r											rw				

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>SRC</b>	1:0	rw	<p><b>Data Source</b></p> <p>The bitfield controls the source of the data to be processed</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> <b>RIFO</b>, Data is pushed from RIF0</li> <li>01<sub>B</sub> <b>RIF1</b>, Data is pushed from RIF1</li> <li>10<sub>B</sub> <b>Both</b>, Data is pushed from RIF0 and RIF1</li> <li>11<sub>B</sub> <b>EMEM</b>, Data will be read from EMEM</li> </ul>
<b>ANT</b>	3:2	rw	<p><b>Number of Antennae</b></p> <p>The number of antennae connected to each RIF instance to be used by this SPU</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> <b>ONE</b>, One Antenna is connected</li> <li>01<sub>B</sub> <b>TWO</b>, Two Antennae are connected</li> <li>10<sub>B</sub> <b>THREE</b>, Three Antennae are connected</li> <li>11<sub>B</sub> <b>FOUR</b>, Four Antennae are connected</li> </ul>
<b>SMPLCNT</b>	14:4	rw	<p><b>Sample Count</b></p> <p>Number of samples per ramp (bin loop count). This field contains the number of data samples per ramp minus one. i.e. for a 512 point data set programme a value of 511.</p>
<b>RES</b>	15, 29:27	r	<b>Reserved</b>
<b>RAMPS</b>	26:16	rw	<p><b>Ramps per Measurement Cycle</b></p> <p>The number of ramps expected to be input for this measurement cycle. The IDM will process this number of ramps and write the data to the buffer memory. Excess data will be silently ignored unless the SPU is retriggered. The number of ramps expected is the contents of this bitfield plus one. i.e. 0 equates to 1 ramp and 2047 equates to 2048 ramps.</p>
<b>SIGNED</b>	30	rw	<p><b>Signed or Unsigned Data</b></p> <p>Defines whether the RIF input data will be treated as unsigned values or two's complement data.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> <b>UNSIGNED</b>, Unsigned Data The Input Data from the RIF is unsigned. This allows 16 bits of resolution. i.e. the input data range is 0 to 2<sup>16</sup>-1</li> <li>1<sub>B</sub> <b>SIGNED</b>, Signed Data The Input Data from the RIF is in 2's complement format</li> </ul>
<b>FORMAT</b>	31	rw	<p><b>RIF Data Format</b></p> <p>RIF data is always 16 bit. This bitfield selects whether the data is real or complex.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> <b>REAL</b>, Input Data is a Real Number</li> <li>1<sub>B</sub> <b>COMPLEX</b>, Input Data has Real and Imaginary Components</li> </ul>

## Signal Processing Unit (SPU)

**Table 635 Access Mode Restrictions of ID\_CONF sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	ANT, FORMAT, RAMPS, SIGNED, SMPLCNT, SRC	
Otherwise (default)	r	ANT, FORMAT, RAMPS, RES, SIGNED, SMPLCNT, SRC	

**Table 636 Reset Values of ID\_CONF**

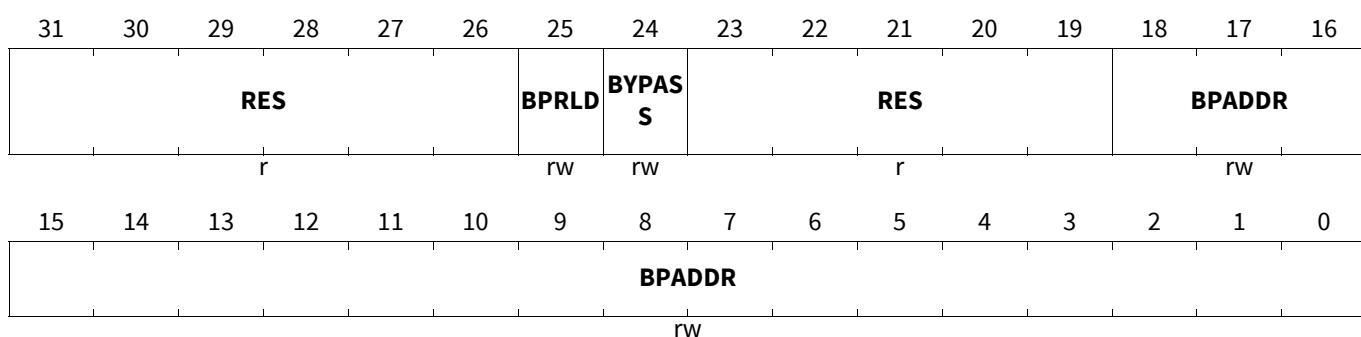
Reset Type	Reset Value	Note
Application Reset	4000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	4000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Input DMA Configuration 2

The Input DMA can bypass data from RIF1 if the incoming data for each ramp exceeds the capacity of the buffer memory. This register configures the bypass mode and also allows the bypassed data to be reloaded into the buffer memory by the Input DMA. Note that, when reloading bypassed data, the data source is still considered to be a RIF for configuration purposes.

#### ID\_CONF2

**Input DMA Configuration 2** (00034<sub>H</sub>) Reset Value: Table 638



## Signal Processing Unit (SPU)

Field	Bits	Type	Description
BPADDR	18:0	rw	<p><b>Bypass Address</b>            This is the base address to be used when reading or writing bypass data. It is a 256 bit (32 byte) aligned address and needs to have five LSBs added to convert to a system address. It allows for a maximum addressable range of 16 MiB. Bypass data is written as a raw data stream as generated by the Radar Interface. No processing is performed by the SPU until the data is read back from the Radar Memory.</p> <p>This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.</p>
RES	23:19, 31:26	r	<b>Reserved</b>
BYPASS	24	rw	<p><b>SPU Bypass Mode</b>            If enabled, and RIF0 and RIF1 are both configured as data sources, then the RIF1 data is written to EMEM directly without processing and can then be reloaded later. Switching between write and read operations is controlled by the BPRLD bitfield. Only the RIF0 data is processed. This mode is only effective for SPU instance 0 and should only be used in a two SPU system if the SPUs are lockstepped so that the data written from SPU1 is ignored.</p>
BPRLD	25	rw	<p><b>Bypass Reload</b>            If set, BYPASS will cause data to be reloaded from Radar Memory starting from BPADDR. If cleared, BYPASS will cause data to be written to Radar Memory starting from BPADDR.</p>

Table 637 Access Mode Restrictions of ID\_CONF2 sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BPADDR, BPRLD, BYPASS	
Otherwise (default)	r	BPADDR, BPRLD, BYPASS, RES	Default Access Mode (Bus Error on Write)

Table 638 Reset Values of ID\_CONF2

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit (SPU)

### Input DMA Configuration: Radar Memory

This register contains options used to configure the Input DMA for reading data from Radar Memory. Register field descriptions should be read in conjunction with [Section 19.3.1.2.1, Principles of Operation \(Radar Memory\)](#) where usage and permissible combinations are more fully defined

#### ID\_RM\_CONF

##### Input DMA Configuration: Radar Memory

(00038<sub>H</sub>)

Reset Value: [Table 640](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16												
<b>PM</b>	<b>BLOCKS</b>		<b>RES</b>	<b>AM</b>		<b>RES</b>	<b>TRNSPS</b>	<b>FORMAT</b>		<b>BASE</b>																	
rw	rw		r	rw		r	rw	rw		rw		rw		rw													
<b>BASE</b>																											
rw																											

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> Base Address of the data to be read from Radar Memory. This is a 32 byte (256 bit) aligned address relative to the start address of the Radar Memory. Five LSBs need to be added to convert to a system address. This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.
<b>FORMAT</b>	21:19	rw	<b>Input Data Format</b> The format of the Input Data to be read from the Radar Memory. Note that the REAL16BIT format must not be used if the TRNSPS bit is also set. 000 <sub>B</sub> <b>CMPLX16BIT</b> , 16 Bit Precision Complex Data 001 <sub>B</sub> <b>CMPLX32BIT</b> , 32 Bit Precision Complex Data 010 <sub>B</sub> <b>PWR32BIT</b> , 32 Bit Precision Power Data 011 <sub>B</sub> <b>RES3</b> , Reserved 100 <sub>B</sub> <b>REAL16BIT</b> , 16 Bit Precision Real Data The format cannot be used if ID_RM_CONF.TRNSPS=1 101 <sub>B</sub> <b>REAL32BIT</b> , 32 Bit Precision Real Data 110 <sub>B</sub> <b>REAL16FP</b> , Half Precision Floating Point, real data The format cannot be used if ID_RM_CONF.TRNSPS=1 111 <sub>B</sub> <b>CMPLX16FP</b> , Half Precision Floating Point, complex data

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
TRNSPS	22	rw	<p><b>Transpose Addressing</b>  This bit should be set when the bins or samples used to construct an FFT input set in the Buffer memory do not occupy contiguous addresses in the Radar Memory. If set, it switches the operating mode of the Input DMA so that each of the bins in the 256 bit word read from radar Memory are treated as bins of different input FFT datasets when being written to buffer memory</p> <p><math>0_B</math> <b>LIN</b>, Linear Mode  Bins stored in contiguous addresses of Radar Memory will be loaded directly into Buffer Memory as an input for an FFT dataset.</p> <p><math>1_B</math> <b>TRN</b>, Transpose  Bins stored in contiguous addresses of Radar Memory will be treated as belonging to different FFT input datasets</p>
RES	23, 27	r	<b>Reserved</b>
AM	26:24	rw	<p><b>Antenna Mapping</b>  Several processing elements of the SPU need to identify each FFT dataset with a particular antenna. This field, in conjunction with the PM field, defines how this should be done. This field applies when PM is set to <math>0_B</math>. Setting PM to <math>1_B</math> is equivalent to setting this field to IDX.</p> <p><math>000_B</math> <b>OFF</b>, Default  Antenna ID passed to processing chain is permanently <math>0_D</math></p> <p><math>001_B</math> <b>IDX</b>, Index Mode  Antenna ID is derived from the dataset index</p> <p><math>010_B</math> <b>ILR</b>, Inner Loop Repeat  Antenna ID is derived from the Inner Loop Repeat Counter Value</p> <p><math>011_B</math> <b>OLR</b>, Outer Loop Repeat  Antenna ID is derived from the Outer Loop Repeat Counter Value</p> <p><math>100_B</math> <b>BLR</b>, Bin Loop Repeat  Antenna ID is derived from the Bin Loop Repeat Counter Value</p> <p><math>101_B</math> <b>RES</b>, Reserved</p> <p>...</p> <p><math>111_B</math> <b>RES</b>, Reserved</p>
BLOCKS	30:28	rw	<p><b>Number of Datablocks</b>  In Integration mode, the number of datablocks to be simultaneously constructed in the buffer memory. Normally when reading across antenna, most of the data read is discarded. If this field is non-zero, the Input DMA Engine will attempt to build multiple datablocks in memory using the additional data. The value of this field is then used to control the maximum number of datablocks that will fit into the physical limit of the buffer memory. The number of datablocks is the field value plus one. The maximum permissible value for this field is 3 for 64 bit data (32 bit precision complex) and 7 for 32 bit data (16 bit precision complex or 32 bit power)</p>

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
PM	31	rw	<p><b>Processing Mode</b></p> <p>This bit is set when the Input DMA is required to build datasets comprising one FFT from each attached antennae. This would be necessary when it is intended to use the coherent or non-coherent integration functions of the MATH2 unit.</p> <p>0<sub>B</sub> <b>DM</b>, Default Mode This is the optimal mode for in-place FFT calculation. All the datapoints read from memory can be directly used to construct a single datablock in the buffer memory. This means that the 32<sub>D</sub> or 64<sub>D</sub> bit data read in each 256<sub>D</sub> bit word will be used by incrementing either the bin loop or inner loop</p> <p>1<sub>B</sub> <b>IM</b>, Integration Mode In Integration Mode, it is assumed that the buffer memory must contain one FFT input dataset for each antenna. Setting this bit forces the buffer memory to be switched only after the completion of a complete pass through the inner loop. If this bit is set, then the Input DMA Engine is allowed to construct multiple datablocks in memory if this is needed to maximise utilisation of the Radar Memory bandwidth. This function is controlled by the ID_RM_CONF.BLOCKS bitfield.</p>

**Table 639 Access Mode Restrictions of ID\_RM\_CONF sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	AM, BASE, BLOCKS, FORMAT, PM, TRNSPS	
Otherwise (default)	r	AM, BASE, BLOCKS, FORMAT, PM, RES, TRNSPS	Default Access Mode (Bus Error on Write)

**Table 640 Reset Values of ID\_RM\_CONF**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Inner Loop Address Offset

Options used to configure the Input DMA for reading data from Radar Memory. Register field descriptions should be read in conjunction with [Section 19.3.1.2.1, Principles of Operation \(Radar Memory\)](#) where usage and permissible combinations are more fully defined

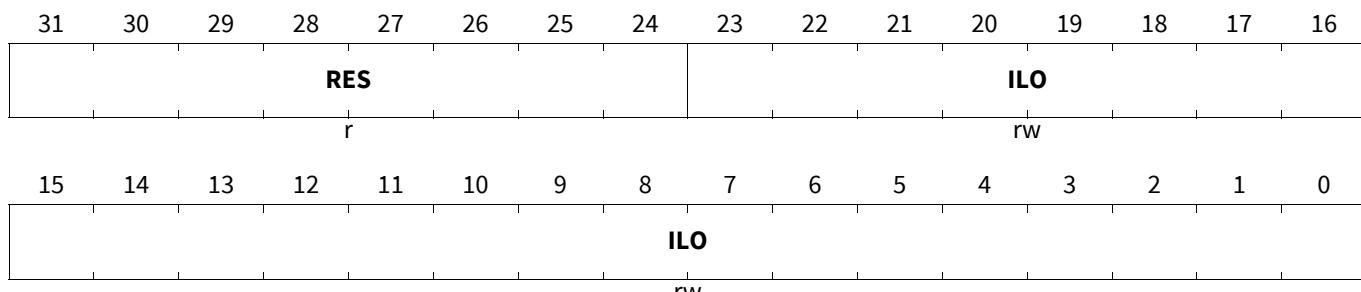
## Signal Processing Unit (SPU)

### ID\_RM\_ILO

Inner Loop Address Offset

(0003C<sub>H</sub>)

Reset Value: [Table 642](#)



Field	Bits	Type	Description
ILO	23:0	rw	<b>Inner Loop Offset</b> The Inner Loop Offset is a byte address. The smallest permissible value is the size of a single data element (e.g. a value of eight for a 32 bit precision complex data word)
RES	31:24	r	<b>Reserved</b>

**Table 641 Access Mode Restrictions of ID\_RM\_ILO sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	ILO	
Otherwise (default)	r	ILO, RES	

**Table 642 Reset Values of ID\_RM\_ILO**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Outer Loop Address Offset

Options used to configure the Input DMA for reading data from Radar Memory. Register field descriptions should be read in conjunction with [Section 19.3.1.2.1, Principles of Operation \(Radar Memory\)](#) where usage and permissible combinations are more fully defined

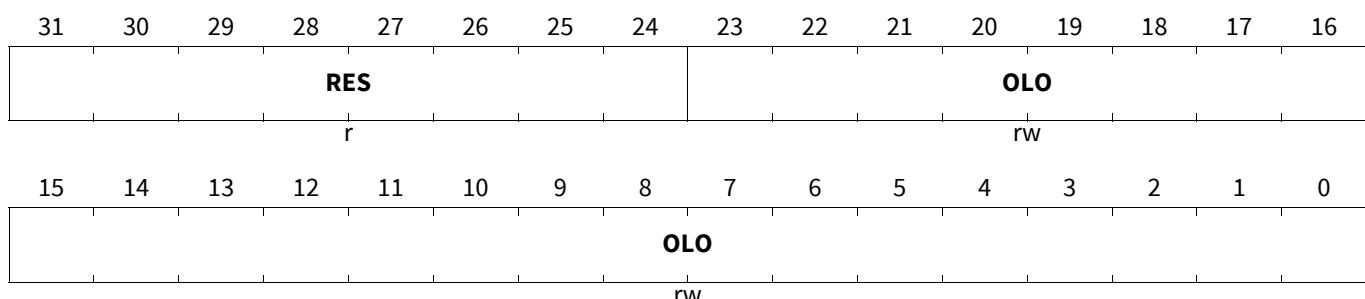
## Signal Processing Unit (SPU)

### ID\_RM\_OLO

Outer Loop Address Offset

(00040<sub>H</sub>)

Reset Value: [Table 644](#)



Field	Bits	Type	Description
OLO	23:0	rw	<b>Outer Loop Offset</b> The Outer Loop Offset is a byte address. The smallest permissible value is the size of a single data element (e.g. a value of eight for a 32 bit precision complex data word)
RES	31:24	r	<b>Reserved</b>

**Table 643 Access Mode Restrictions of ID\_RM\_OLO sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	OLO	
Otherwise (default)	r	OLO, RES	

**Table 644 Reset Values of ID\_RM\_OLO**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Bin Offset Address Configuration

Options used to configure the Input DMA for reading data from Radar Memory. Register field descriptions should be read in conjunction with [Section 19.3.1.2.1, Principles of Operation \(Radar Memory\)](#) where usage and permissible combinations are more fully defined

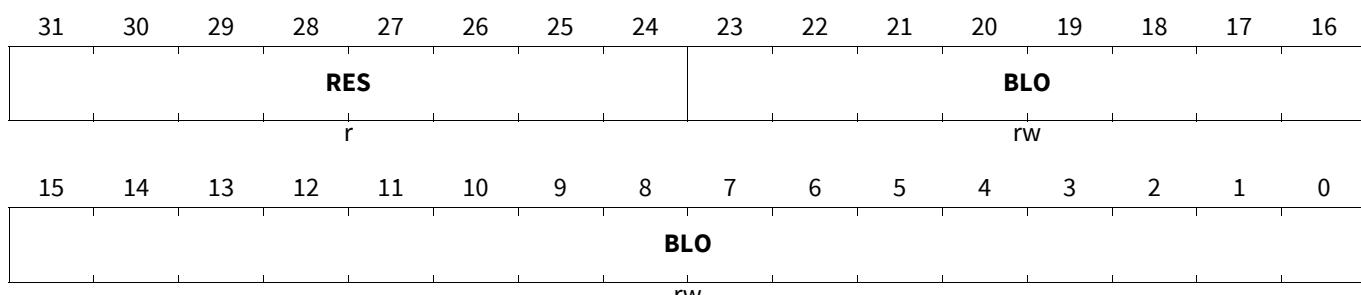
## Signal Processing Unit (SPU)

### ID\_RM\_BLO

#### Bin Offset Address Configuration

(00044<sub>H</sub>)

Reset Value: [Table 646](#)



Field	Bits	Type	Description
BLO	23:0	rw	<b>Bin Loop Offset</b> The Bin Loop Offset is a byte address. The smallest permissible value is the size of a single data element (e.g. a value of eight for a 32 bit precision complex data word)
RES	31:24	r	<b>Reserved</b>

**Table 645 Access Mode Restrictions of ID\_RM\_BLO sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BLO	
Otherwise (default)	r	BLO, RES	

**Table 646 Reset Values of ID\_RM\_BLO**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Inner and Outer Loop Repeat

Options used to configure the Input DMA for reading data from Radar Memory. The fields in this register define the number of Iterations for the Inner and Outer Addressing Loops. Register field descriptions should be read in conjunction with [Section 19.3.1.2.1, Principles of Operation \(Radar Memory\)](#) where usage and permissible combinations are more fully defined

## Signal Processing Unit (SPU)

### ID\_RM\_IOLR

Inner and Outer Loop Repeat

(00048<sub>H</sub>)

Reset Value: [Table 648](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		OLR								rw					
RES		ILR								rw					

Field	Bits	Type	Description
ILR	12:0	rw	<b>Inner Loop Repeat</b> Repetition Count for the Inner Address Loop. The loop will repeat at least once. The overall number of repetitions will be the value of this bitfield + 1.
RES	15:13, 31:29	r	<b>Reserved</b>
OLR	28:16	rw	<b>Outer Loop Repeat</b> Repetition Count for the Outer Address Loop. The loop will repeat at least once. The overall number of repetitions will be the value of this bitfield + 1.

**Table 647 Access Mode Restrictions of ID\_RM\_IOLR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	ILR, OLR	
Otherwise (default)	r	ILR, OLR, RES	Default Access Mode (Bus Error on Write)

**Table 648 Reset Values of ID\_RM\_IOLR**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Bin Loop Repeat

Options used to configure the Input DMA for reading data from Radar Memory. The field in this register defines the number of Iterations for the Bin Addressing Loop. Register field descriptions should be read in conjunction with [Section 19.3.1.2.1, Principles of Operation \(Radar Memory\)](#) where usage and permissible combinations are more fully defined

**Signal Processing Unit (SPU)****ID\_RM\_BLR****Bin Loop Repeat****(0004C<sub>H</sub>)****Reset Value: Table 650**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				BLR											
r				rw											

Field	Bits	Type	Description
<b>BLR</b>	12:0	rw	<b>Bin Loop Repeat</b> Repetition Count for the Bin Address Loop. The loop will repeat at least once. The overall number of repetitions will be the value of this bitfield + 1.
<b>RES</b>	31:13	r	<b>Reserved</b>

**Table 649 Access Mode Restrictions of ID\_RM\_BLR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BLR	
Otherwise (default)	r	BLR, RES	Default Access Mode (Bus Error on Write)

**Table 650 Reset Values of ID\_RM\_BLR**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Spare Configuration Register****ID\_RM\_ACFG0****Spare Configuration Register****(00050<sub>H</sub>)****Reset Value: Table 652**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES															
r															

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>RES</b>	31:0	r	<b>Reserved</b>

**Table 651 Access Mode Restrictions of ID\_RM\_ACFG0 sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)

**Table 652 Reset Values of ID\_RM\_ACFG0**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Spare Configuration Register

### ID\_RM\_ACFG1

<b>Spare Configuration Register</b>																<b>(00054<sub>H</sub>)</b>				<b>Reset Value: Table 654</b>				
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16									
																<b>RES</b>								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0									
																<b>RES</b>								

Field	Bits	Type	Description
<b>RES</b>	31:0	r	<b>Reserved</b>

**Table 653 Access Mode Restrictions of ID\_RM\_ACFG1 sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)

## Signal Processing Unit (SPU)

**Table 654 Reset Values of ID\_RM\_ACFG1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Partial-Acquisition Counter

This counter is used to monitor and control partial-acquisition. If enabled, it will monitor the input data stream from the RIF and increment at the end of each ramp as determined by the sample count and number of antennae captured from the IDM configuration when the counter was last reset. As the counter increments at the end of the ramp, the first ramp of the measurement cycle will be considered as ramp 0.

Note: LDMST should be used only with bit mask enabled for the 'rh' bits in this register. Hardware updates occurring between the read and write phases of an RMW may be overwritten.

### PACTR

**Partial-Acquisition Counter** Reset Value: [Table 656](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
<b>RES</b>								<b>COUNT</b>								
r								rh								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
<b>ATTN</b>	<b>ERR</b>	<b>TRIG</b>	<b>LIMIT</b>												<b>EN</b>	<b>RST</b>
rw	rw	rw	rw								rwh				w	

Field	Bits	Type	Description
<b>RST</b>	0	w	<b>Counter Reset</b> Write 1 to reset the Counter and Update the Control Information. Always reads as 0
<b>EN</b>	1	rwh	<b>Counter Enable</b> This field is used to enable the counter. It will only be updated if the counter is reset on the same write <b>0<sub>B</sub></b> <b>DISABLE</b> , Disable the Counter <b>1<sub>B</sub></b> <b>ENABLE</b> , Enable the Counter
<b>LIMIT</b>	12:2	rw	<b>Preacquisition Counter Limit</b> Limit value for the counter. For purposes of comparison, the first ramp of a measurement cycle is considered to be ramp 0
<b>TRIG</b>	13	rw	<b>Trigger on Limit</b> If this bit is set, then the SPU will start acquiring data when the counter reaches the value in the LIMIT field. This is considered as an internal trigger event and the CTRL.MODE bitfield should be set for internal trigger.

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
ERR	14	rw	<b>Error Interrupt on Limit</b> If this bit is set an error interrupt will be triggered if the counter reaches the value in the LIMIT field
ATTN	15	rw	<b>Attention Request Interrupt onLimit</b> If this bit is set an attention request interrupt will be triggered if the counter reaches the value in the LIMIT field
COUNT	26:16	rh	<b>Counter Value</b> The current value of the Partial Acquisition Counter
RES	31:27	r	<b>Reserved</b> Always returns 0 when read

**Table 655 Access Mode Restrictions of PACTR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rh	COUNT	
	rw	ATTN, ERR, LIMIT, TRIG	
	rwh	EN	
	w	RST	
Otherwise (default)	r	ATTN, ERR, LIMIT, RES, TRIG	Default Access Mode (Bus Error on Write)
	rX	RST	
	rh	COUNT, EN	

**Table 656 Reset Values of PACTR**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

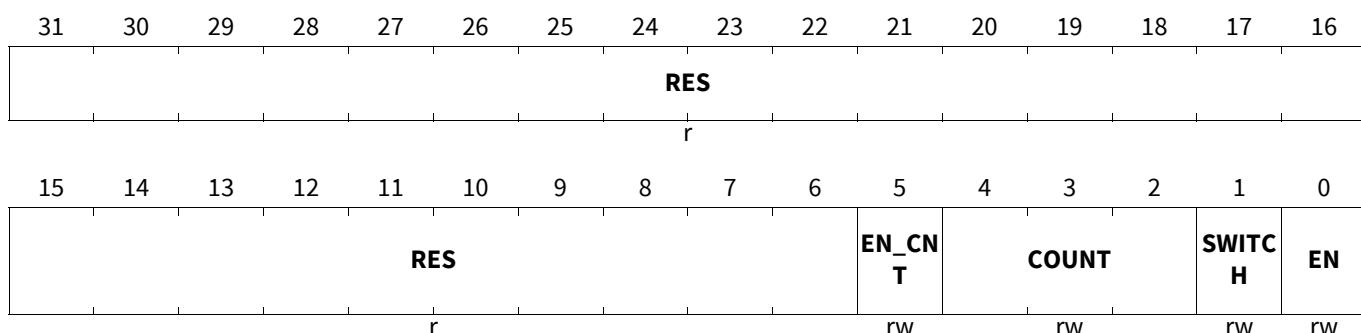
### Double Pass Configuration

Double Pass Mode allows the configuration of the MATH1 unit to be changed during processing of ADC data. The selected configuration can be alternated on a ramp-by ramp basis or both programmed configurations can be applied to all ramps.

## Signal Processing Unit (SPU)

### DPASS\_CONF

#### Double Pass Configuration

(0005C<sub>H</sub>)Reset Value: [Table 658](#)

Field	Bits	Type	Description
<b>EN</b>	0	rw	<b>Enable</b> The SPU allows all processing operations to be run twice per data block or different processing parameters to be used on sequences of data blocks. This mode is enabled using the DPASS_CONF.EN bitfield. This results in two sets of results being written to Radar Memory
<b>SWITCH</b>	1	rw	<b>Buffer Memory Switch</b> If set to 0 <sub>B</sub> , both configurations of the MATH1 and MATH2 units will be applied to all data blocks and two full sets of output data will be written to the Radar Memory. If set to 1 <sub>B</sub> , the two configurations for MATH1 and MATH2 will be applied to alternate data blocks. Each set of output data written to Radar memory will contain half the data.
<b>COUNT</b>	4:2	rw	<b>Switch Count</b> If DPASS_CONF.EN_CNT is set, this field determines the number of data blocks to be processed before the alternate sets of window parameters are switched. The number of data blocks processed by the Loader between parameter switches is COUNT+1. i.e. if COUNT=0 then the window parameters will be switched for every data block processed, if COUNT=7, then the parameters will be switched after every eight data blocks processed.
<b>EN_CNT</b>	5	rw	<b>Enable Count</b> This bit enables switching the two sets of window parameters based on the DPASS_CONF.COUNT bitfield rather than the "Double Pass" settings. When set, the window parameters are switched every time a counter counting the number of data blocks processed reached the terminal count stored in the COUNT field + 1. The DPASS_CONF.EN and DPASS_CONF.SWITCH settings have no effect on the window parameters when this bit is set.
<b>RES</b>	31:6	r	<b>Reserved</b>

## Signal Processing Unit (SPU)

**Table 657 Access Mode Restrictions of DPASS\_CONF sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	COUNT, EN, EN_CNT, SWITCH	
Otherwise (default)	r	COUNT, EN, EN_CNT, RES, SWITCH	Default Access Mode (Bus Error on Write)

**Table 658 Reset Values of DPASS\_CONF**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Loader Configuration

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Configuration Parameters for the Loader and MATH1 components of Streaming Processor 1

#### BEx\_LDR\_CONF (x=0-1)

**Loader Configuration** **Reset Value:** [Table 660](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>IFFT</b>	<b>FFTBY PS</b>	<b>FORMAT</b>			<b>SIZE</b>		<b>PHSHFT</b>					<b>EXPNT</b>			
rw	rw	rw			rw		rw					rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>DRPF</b>								<b>DRPL</b>							
rw								rw							

Field	Bits	Type	Description
<b>DRPL</b>	7:0	rw	<b>Drop Last</b> Stores a number "m". The last m samples of the input data will be ignored
<b>DRPF</b>	15:8	rw	<b>Drop First</b> Stores a number "n". The first n samples of the input data will be ignored

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>EXPNT</b>	21:16	rw	<b>Common Exponent</b> For $16_D$ bit data, this indicates the alignment correction to be applied when reformatting to 32 bit. It should be set to $0_D$ if no adjustment of magnitude is needed. Otherwise the $16_D$ bit data will be shifted left by the number of bits set in this field and sign extended to $32_D$ bits. Empty bits at the right end of the $32_D$ bit value will be set to $0_B$ . The maximum expected value for this field is $16_D$ when using data read from Radar Memory as this is the largest value that will be generated using the compression function in the output data manager. Larger values can be used to create some gain in the input signal. Overflows in the shift operation will cause saturation of the output.
<b>PHSHFT</b>	23:22	rw	<b>Fast Phase Shift</b> This bitfield can set a fast phase shift of 0, 90, 180 or 270 degrees $00_B$ <b>ZERO</b> , No Phase Shift $01_B$ <b>NINETY</b> , 90 degree phase shift $10_B$ <b>ONEEIGHTY</b> , 180 degree phase shift $11_B$ <b>TWOSEVENTY</b> , 270 degree phase shift
<b>SIZE</b>	27:24	rw	<b>FFT Size</b> Number of samples for the FFT. Must be a power of two between 8 and 2048. Unspecified values are reserved and must not be written. $0_H$ <b>8</b> , FFT is eight data samples $1_H$ <b>16</b> , FFT is sixteen data samples $2_H$ <b>32</b> , FFT is thirty-two data samples $3_H$ <b>64</b> , FFT is sixty-four data samples $4_H$ <b>128</b> , FFT is one hundred and twenty-eight data samples $5_H$ <b>256</b> , FFT is two hundred and fifty-six data samples $6_H$ <b>512</b> , FFT is five hundred and twelve data samples $7_H$ <b>1024</b> , FFT is one thousand and twenty-four data samples $8_H$ <b>2048</b> , FFT is two thousand and forty-eight data samples
<b>FORMAT</b>	29:28	rw	<b>Window Data Format</b> Format of the Window Function Data stored in the configuration memory. If a complex format is selected, the real component of each value is always stored at the lower address in memory. $00_B$ <b>REAL16</b> , 16 Bit Real Window Data $01_B$ <b>REAL32</b> , 32 Bit Real Window Data $10_B$ <b>CMPLX16</b> , 16 Bit Complex Window Data $11_B$ <b>CMPLX32</b> , 32 Bit Complex Window Data
<b>FFTBYPS</b>	30	rw	<b>FFT Bypass</b> Bypass the FFT Module
<b>IFFT</b>	31	rw	<b>Inverse FFT Enable</b> If set, the accelerator will perform an IFFT operation instead of an FFT. In IFFT mode, the automatic descaling of the FFT output is disabled

## Signal Processing Unit (SPU)

**Table 659 Access Mode Restrictions of `BEx_LDR_CONF` (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	DRPF, DRPL, EXPNT, FFTBYP, FORMAT, IFFT, PSHFT, SIZE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	DRPF, DRPL, EXPNT, FFTBYP, FORMAT, IFFT, PSHFT, SIZE	Default Access Mode (Bus Error on Write)

**Table 660 Reset Values of `BEx_LDR_CONF` (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Loader Configuration Extended

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Configuration Parameters for the Loader and MATH1 components of Streaming Processor 1

#### **BEx\_LDR\_CONF2 (x=0-1)**

Loader Configuration Extended

(00064<sub>H</sub>+x\*40<sub>H</sub>)

Reset Value: [Table 662](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>WINEN</b>	<b>RES</b>		<b>PADF</b>												
rw	r								rw						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>WBASE</b>															
									rw						

Field	Bits	Type	Description
<b>WBASE</b>	15:0	rw	<b>Window Coefficient Base Address</b> This is the base address of the Window Coefficient Array in the Configuration Memory. This address is relative to the Configuration Memory Base Address and must be 8 byte aligned. Any non-zero value written to bits[2:0] will be ignored.
<b>PADF</b>	28:16	rw	<b>Pad at Front</b>
<b>RES</b>	30:29	r	<b>Reserved</b>
<b>WINEN</b>	31	rw	<b>Window Function Enable</b> 0 <sub>B</sub> <b>OFF</b> , Disable Window Function 1 <sub>B</sub> <b>ON</b> , Enable Window Function

## Signal Processing Unit (SPU)

**Table 661 Access Mode Restrictions of BEx\_LDR\_CONF2 (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	PADF, WBASE, WINEN	
Otherwise (default)	r	PADF, RES, WBASE, WINEN	Default Access Mode (Bus Error on Write)

**Table 662 Reset Values of BEx\_LDR\_CONF2 (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Antenna Offset**

Offset Addresses for each antenna relative to the base address of the windowing coefficients. Each antenna has a unique offset stored as a 16 bit address. This address must be 8 byte aligned. Any non-zero value written to bits[2:0] will be ignored.

**BEx\_Aj\_ANTOFST (j=0-3;x=0-1)**

**Antenna Offset** **(00068<sub>H</sub>+x\*40<sub>H</sub>+j\*4)** **Reset Value: Table 664**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADROFST1_ANT															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADROFST0_ANT															
rw															

Field	Bits	Type	Description
ADROFST <sub>i</sub> _AN T (i=0-1)	16*i+15:16*i	rw	Antenna Offset Address

**Table 663 Access Mode Restrictions of BEx\_Aj\_ANTOFST (j=0-3;x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ADROFST <sub>i</sub> _ANT (i=0-1)	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ADROFST <sub>i</sub> _ANT (i=0-1)	Default Access Mode (Bus Error on Write)

## Signal Processing Unit (SPU)

**Table 664 Reset Values of BEx\_Aj\_ANTOFST (j=0-3;x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Unloader Configuration

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Configuration Parameters for the Power Histogram Unit contained in the Unloader. This register also defines the format of the data to be written to the buffer memory.

#### BEx\_UNLDR\_CONF (x=0-1)

**Unloader Configuration** (00078<sub>H</sub>+x\*40<sub>H</sub>) Reset Value: Table 666

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>HISTBASE</b>															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HISTBINS		RES		AFV			HAFE		HISTE N		FORMAT		EXPNT		
rw		r		rw			rw		rw		rw		rw		

Field	Bits	Type	Description
<b>EXPNT</b>	4:0	rw	<b>Common Exponent</b> When writing 16 bit precision data to the buffer memory, this field indicates the number of LSBs to be removed. i.e. the 32 bit data will be shifted right by this number of bit positions before truncating to the 16 least significant bits. In this mode, the value of the field should be restricted to values between 0 <sub>D</sub> and 16 <sub>D</sub> . The results of setting a value greater than 16 <sub>D</sub> are undefined. When writing 32 bit precision data to the buffer memory, this field indicates the number of LSBs to be added. i.e. the 32 bit data will be shifted left by this number of bit positions. The remaining bits will be set to maximum or minimum value if overflow or underflow has occurred. The precision of the output data will be determined by the FORMAT bitfield.
<b>FORMAT</b>	5	rw	<b>Input Data Format</b> Required format of the data to be written to the buffer memory. Set to 1, 32 bit precision, after reset. 0 <sub>B</sub> <b>16BIT</b> , 16 Bit Complex Data 1 <sub>B</sub> <b>32BIT</b> , 32 Bit Complex Data

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
HISTEN	6	rw	<b>Histogram Enable</b> If zero, no histogram is generated. If 1, then a power histogram is cumulatively assembled until the mode is disabled again
HAFE	7	rw	<b>Histogram Antenna Filter Enable</b> Use this control bit to enable filtering of the data used to generate the power histogram based on antenna ID. If filtering is enabled only the selected antenna will contribute to the histogram data and the counter used for STRT and END check will only be incremented when an FFT associated that antenna ID is processed. 0 <sub>B</sub> OFF, Antenna filtering is disabled 1 <sub>B</sub> ON, Antenna filtering is enabled
AFV	10:8	rw	<b>Antenna Filter Value</b> If HAFE is set, this will be the antenna index to use to accumulate the power histogram data. Only data from this antenna index will be used to update the power histogram
RES	12:11	r	<b>Reserved</b>
HISTBINS	15:13	rw	<b>Number of power values per histogram bin.</b> Number of LSBs of power value to ignore when calculating histogram bin to increment. The calculated power value will be shifted right by the value of this bitfield and the resultant number used as the relative address of the histogram bin to increment.
HISTBASE	31:16	rw	<b>Histogram Base Address</b> Base address in the configuration RAM of the power histogram. This address is relative to the configuration RAM base address and must be 8 byte aligned. Any non-zero value written to bits[2:0] will be ignored.

Table 665 Access Mode Restrictions of BEx\_UNLDR\_CONF (x=0-1) sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	AFV, EXPNT, FORMAT, HAFE, HISTBASE, HISTBINS, HISTEN	
Otherwise (default)	r	AFV, EXPNT, FORMAT, HAFE, HISTBASE, HISTBINS, HISTEN, RES	Default Access Mode (Bus Error on Write)

Table 666 Reset Values of BEx\_UNLDR\_CONF (x=0-1)

Reset Type	Reset Value	Note
Application Reset	0000 0020 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0020 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit (SPU)

### Unloader Configuration 2

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

**BEx\_UNLDR\_CONF2 (x=0-1)**

**Unloader Configuration 2**

(0007C<sub>H</sub>+x\*40<sub>H</sub>)

**Reset Value: Table 668**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
END															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STRT															
rw															

Field	Bits	Type	Description
<b>STRT</b>	15:0	rw	<b>Start Count</b> Delay from Start of Measurement Cycle before the accumulation of histogram is enabled. If set to 0 accumulation will start with the first FFT of the measurement cycle
<b>END</b>	31:16	rw	<b>End Count</b> Delay from Start of Measurement Cycle before the accumulation of histogram is ended. If set to all 1s accumulation will end with the final FFT of the measurement cycle

**Table 667 Access Mode Restrictions of BEx\_UNLDR\_CONF2 (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	END, STRT	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	END, STRT	Default Access Mode (Bus Error on Write)

**Table 668 Reset Values of BEx\_UNLDR\_CONF2 (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Spare Configuration Register

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Reserved for future expansion of the Unloader functionality

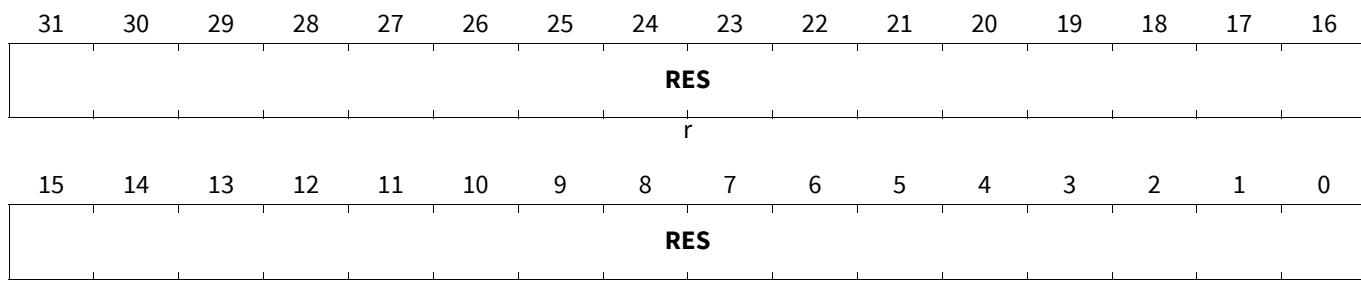
## Signal Processing Unit (SPU)

### BEx\_UNLDR\_ACFG (x=0-1)

#### Spare Configuration Register

( $00080_H + x * 40_H$ )

Reset Value: [Table 670](#)



Field	Bits	Type	Description
RES	31:0	r	Reserved

**Table 669 Access Mode Restrictions of BEx\_UNLDR\_ACFG (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)

**Table 670 Reset Values of BEx\_UNLDR\_ACFG (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	$0000\ 0000_H$	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	$0000\ 0000_H$	Kernel Reset (software controlled by KRST0-1 registers)
Application Reset	$0000\ 0000_H$	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	$0000\ 0000_H$	Kernel Reset (software controlled by KRST0-1 registers)

### Output Data Processor Configuration

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

This register configures the Output Data Processor for writing FFT results to the Radar Memory.

## Signal Processing Unit (SPU)

**BEx\_ODP\_CONF (x=0-1)****Output Data Processor Configuration**(00084<sub>H</sub>+x\*40<sub>H</sub>)Reset Value: [Table 672](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>RES</b>	<b>HPFP</b>	<b>ROF</b>	<b>IPF</b>				<b>EXPNT</b>			<b>SCALE</b>	<b>FTR</b>	<b>MODE</b>			<b>BASE</b>
r	rw	rw	rw				rw			rw	rw	rw			rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>BASE</b>															
rw															

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> The base address to be used when writing data to Radar Memory. This is a word (256 bit) address relative to the Radar Memory base address. All writes will be 32 byte words. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing an FFT, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.
<b>MODE</b>	19	rw	<b>ODP Mode</b> Major Operating Mode for the ODP 0 <sub>B</sub> <b>OFF</b> , FFT Output Disable The FFT output path is disabled. 1 <sub>B</sub> <b>ON</b> , FFT On The FFT data output path is enabled
<b>FTR</b>	20	rw	<b>Force to Real</b> The complex component of the FFT output data will be set to zero when it is read from the buffer memory. This will affect all processing operations of the MATH2 unit.
<b>SCALE</b>	21	rw	<b>Scale Results to 16 bit</b> If set, the results will be scaled to 16 bit precision before writing to EMEM. The scaling factor used will be from the ODP_CONF.EXPNT bitfield 0 <sub>B</sub> <b>32BIT</b> , Output is written with 32 bit precision 1 <sub>B</sub> <b>16BIT</b> , Output is written with 16 bit precision
<b>EXPNT</b>	26:22	rw	<b>Common Exponent</b> When writing 16 bit precision data to the Radar Memory, this field indicates the number of LSBs to be removed. i.e. the 32 bit data will be shifted right by this number of bit positions before truncating to the 16 least significant bits. The remaining bits will be set to maximum or minimum value if overflow or underflow has occurred

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
IPF	27	rw	<b>In Place FFT</b> When set, the Output Data Processor will attempt to write FFT results to the memory locations freed up by the reading of the input data.
ROF	28	rw	<b>Real Only Format</b> When set, the Output Data Processor will write FFT output data in a real only format. The imaginary component of the results will not be written. This mode differs from the function of "Force to Real" (FTR) because any other calculations performed using the FFT data will use the imaginary component. If consistency is required between the real only data written and the results of the other channels, then FTR should also be set.
HPFP	29	rw	<b>Half Precision Floating Point</b> When set, the Output Data Processor will write FFT output data in half precision floating point format. If this bit is set then SCALE is redundant and will be ignored. FTR and ROF can still be used.
RES	31:30	r	<b>Reserved</b>

**Table 671 Access Mode Restrictions of BEx\_ODP\_CONF (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BASE, EXPNT, FTR, HPFP, IPF, MODE, ROF, SCALE	
Otherwise (default)	r	BASE, EXPNT, FTR, HPFP, IPF, MODE, RES, ROF, SCALE	Default Access Mode (Bus Error on Write)

**Table 672 Reset Values of BEx\_ODP\_CONF (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### NCI Control

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Configuration Parameters for the Non-Coherent Integration Unit and Base address for writing the data to Radar Memory.

## Signal Processing Unit (SPU)

**BEx\_NCICTRL (x=0-1)**

NCI Control

(00088<sub>H</sub>+x\*40<sub>H</sub>)Reset Value: [Table 674](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES					SCALE		FORMAT		EN		BASE				
r					rw		rw	rw	rw	rw	rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE					rw										

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> The base address to be used when writing data to Radar Memory. This is a word (256 bit) address relative to the Radar Memory base address. All writes will be 32 byte words. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing an FFT, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.
<b>EN</b>	19	rw	<b>Enable</b> Enable the Non-Coherent Integration Unit. This must be set if the CFARCTRL.CFAREN bitfield is set to CFARN as the CFAR unit then requires NCI unit output. If this condition is not met the CFAR operation will be undefined as the input data will not be as expected.
<b>FORMAT</b>	21:20	rw	<b>Output Format</b> Sets the precision of the output data written to the Radar Memory. 00 <sub>B</sub> <b>OFF</b> , Output Channel Off The output from the Non-coherent integration unit is available for use by the CFAR module but is not written to Radar Memory 01 <sub>B</sub> <b>REAL16BIT</b> , 16 Bit Data Output (real) 10 <sub>B</sub> <b>REAL32BIT</b> , 32 Bit Data Output (real) 11 <sub>B</sub> <b>CMPLX16BIT</b> , 16 Bit Precision Complex Data Output CMPLX16BIT. 16 bit precision, complex data will be written with a zero, imaginary component. The complex data mode is to allow the data to be read back into the SPU for further processing
<b>SCALE</b>	23:22	rw	<b>Result Scaling</b> This bitfield controls the scaling factor applied to the results 00 <sub>B</sub> <b>OFF</b> , No Rescaling 01 <sub>B</sub> <b>DIV2</b> , Divide Result by 2 10 <sub>B</sub> <b>DIV4</b> , Divide Result by 4 11 <sub>B</sub> <b>DIV8</b> , Divide Result by 8
<b>RES</b>	31:24	r	<b>Reserved</b>

## Signal Processing Unit (SPU)

**Table 673 Access Mode Restrictions of BEx\_NCICTRL (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BASE, EN, FORMAT, SCALE	
Otherwise (default)	r	BASE, EN, FORMAT, RES, SCALE	Default Access Mode (Bus Error on Write)

**Table 674 Reset Values of BEx\_NCICTRL (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Summation Unit Control

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Controls the operation of the Summation Unit and sets the base address for writing the data to Radar Memory.

#### BEx\_SUMCTRL (x=0-1)

#### Summation Unit Control

(0008C<sub>H</sub>+x\*40<sub>H</sub>)

Reset Value: [Table 676](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>RES</b>	<b>USEANT</b>							<b>REAL</b>	<b>SUMMODE</b>	<b>PWRM ODE</b>	<b>BASE</b>				
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>BASE</b>															
rw															

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<p><b>Radar Memory Base Address</b></p> <p>The base address to be used when writing data to Radar Memory. This is a word (256 bit) address relative to the Radar Memory base address. All writes will be 32 byte words. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If SUMMODE is SUMANT, if a partial word remains at the end of processing an FFT, the data will be padded to 32 bytes and flushed to Radar Memory. For other modes, the data will be padded to 32 bytes and flushed if a partial word remains at the end of a measurement cycle. This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.</p>
<b>PWRMODE</b>	19	rw	<p><b>Power Mode</b></p> <p>Operating Mode of the Summation Units functions operating in the power domain</p> <p>0<sub>B</sub> <b>OFF</b>, Power Summation is disabled 1<sub>B</sub> <b>SUM</b>, Power Summation is Enabled</p>
<b>SUMMODE</b>	21:20	rw	<p><b>Summation Mode</b></p> <p>Operating Mode of the Summation Unit functions operating on complex or linear power data</p> <p>00<sub>B</sub> <b>OFF</b>, Disabled 01<sub>B</sub> <b>SUM</b>, Sum Complex Data Sum all the complex values in an FFT result if SUMCTRL.REAL = 0 else if SUMCTRL.REAL = 1, set the complex component to 0 before summing so that the output consists of a real component only. Output one datapoint per FFT 10<sub>B</sub> <b>SUMLINP</b>, Sum Linear Power Sum of the linear power calculated from each datapoint in an FFT result. Outputs one data value per FFT processed 11<sub>B</sub> <b>SUMANT</b>, Sum Across Antenna</p>
<b>REAL</b>	22	rw	<p><b>Real Value</b></p> <p>Calculations Using Complex Data will use the real component only. The imaginary component will be set to zero before the calculation. No imaginary component will be written to memory. The field qualifies operation when SUMCTRL.SUMMODE=SUM or SUMCTRL.SUMMODE=SUMANT. It has no effect for other operating modes.</p> <p>0<sub>B</sub> <b>COMPLEX</b>, Complex Arithmetic Calculations will use real and imaginary components 1<sub>B</sub> <b>REAL</b>, Real Arithmetic Only Calculations will use the real component of the input data only. The imaginary component will be discarded. Data written to memory will be a real number</p>

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>USEANT</b>	30:23	rw	<b>Antennae to Use</b> When using the SUMMODE=SUMANT, this field allows the application to select which antennae are to be used in the sum. Each bit in the field corresponds to one antenna with the LSB mapping to antenna 0 and the MSB to antenna 7. If a bit is set to 1, then the relevant antenna will be included in the sum.
<b>RES</b>	31	r	<b>Reserved</b> Always reads as 0

**Table 675 Access Mode Restrictions of BEx\_SUMCTRL (x=0-1) sorted by descending priority**

<b>Mode Name</b>	<b>Access Mode</b>		<b>Description</b>
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BASE, PWRMODE, REAL, SUMMODE, USEANT	
Otherwise (default)	r	BASE, PWRMODE, REAL, RES, SUMMODE, USEANT	Default Access Mode (Bus Error on Write)

**Table 676 Reset Values of BEx SUMCTRL (x=0-1)**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Power Summation

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

This register sets the Base Address in Radar Memory for summed  $\log_2$  power data written when SUMCTRL.PWRMODE=SUM as well as providing additional options for the SUMCTRL.SUMMODE=SUMANT mode.

**BEx\_PWRSUM (x=0-1)**

## Power Summation

(00090<sub>H</sub>+x\*40<sub>H</sub>)

## **Reset Value: Table 678**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16			
<b>RES</b>												<b>PRECISION</b>	<b>SCALE</b>			<b>BASE</b>		
r													rw			rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
<b>BASE</b>																		
rw																		

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> The base address to be used when writing data to Radar Memory. This is a word (256 bit) address relative to the Radar Memory base address. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing a measurement cycle, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.
<b>SCALE</b>	20:19	rw	<b>Sum Antenna Result Scaling</b> This bitfield will control the scaling factor applied to the results of the "sum antenna" operation (SUMCTRL.SUMMODE=SUMANT). The scaling factor can be used to prevent an overflow of the results. 00 <sub>B</sub> <b>OFF</b> , No Rescaling 01 <sub>B</sub> <b>DIV2</b> , Divide Result by 2 10 <sub>B</sub> <b>DIV4</b> , Divide Result by 4 11 <sub>B</sub> <b>DIV8</b> , Divide Result by 8
<b>PRECISION</b>	21	rw	<b>Output Precision</b> Sets the precision of the output data of the "sum antennae" operation written to the Radar Memory when SUMCTRL.SUMMODE=SUMANT. 0 <sub>B</sub> <b>CMPLX32BIT</b> , 32 Bit Precision Complex Data Output 1 <sub>B</sub> <b>CMPLX16BIT</b> , 16 Bit Precision Complex Data Output
<b>RES</b>	31:22	r	<b>Reserved</b> Always reads 0

**Table 677 Access Mode Restrictions of BEx\_PWRSUM (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BASE, PRECISION, SCALE	
Otherwise (default)	r	BASE, PRECISION, RES, SCALE	Default Access Mode (Bus Error on Write)

**Table 678 Reset Values of BEx\_PWRSUM (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit (SPU)

### Power Information Channel Control

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Enables the Output DMA channel writing log<sub>2</sub> power information to the Radar Memory and sets the base address for writes.

#### BEx\_PWRCTRL (x=0-1)

**Power Information Channel Control** **(00094<sub>H</sub>+x\*40<sub>H</sub>)** **Reset Value: Table 680**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								RES							
rw						r									rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								BASE							
									rw						

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> The base address to be used when writing data to Radar Memory. This is a word (256 bit) address relative to the Radar Memory base address. All writes will be 32 byte words. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing an FFT, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.
<b>RES</b>	30:19	r	<b>Reserved</b>
<b>EN</b>	31	rw	<b>Enable</b> Enables the writing of Signal Power to Radar Memory on a per FFT bin basis. The data is a 16 bit precision representation of log2 power.

**Table 679 Access Mode Restrictions of BEx\_PWRCTRL (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BASE, EN	
Otherwise (default)	r	BASE, EN, RES	Default Access Mode (Bus Error on Write)

## Signal Processing Unit (SPU)

**Table 680 Reset Values of BEx\_PWRCTRL (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### CFAR Module Control

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

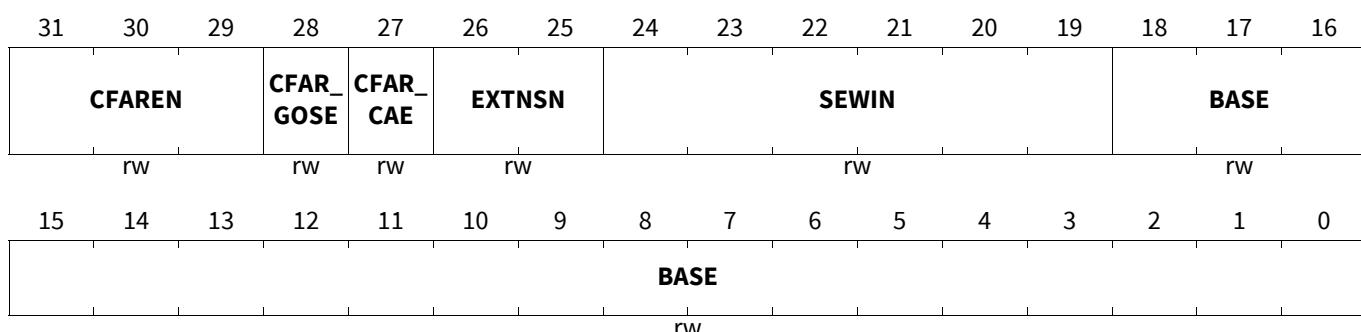
This register controls the operating mode of the CFAR module and the alternative Threshold Unit.

**BEx\_CFARCTRL (x=0-1)**

**CFAR Module Control**

(00098<sub>H</sub>+x\*40<sub>H</sub>)

**Reset Value: Table 682**



Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>CFAR Base Address</b> Base Address in the EMEM to be used for writing the CFAR information. All writes will be 32 byte words. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing a measurement cycle, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.
<b>SEWIN</b>	24:19	rw	<b>Spectrum Extension Window</b> This bitfield defines the window size to be used for spectrum extension. For use with the Local Maximum Unit, the permitted values are 1 or 2 (i.e. a window size of 1 or 2 bins either side of the Bin Under Test). For use with the CFAR module, the window size should be set to the maximum required by the active CFAR algorithm or algorithms. This field must not be set to a non-zero value unless either the CFAR or Local Maximum Unit is enabled.

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>EXTNSN</b>	26:25	rw	<p><b>Spectrum Extension</b></p> <p>Enable Spectrum Extension in either range or velocity modes for the CFAR or Threshold Units. This field must be set to "OFF" unless either the CFAR or Local Maximum Unit is enabled.</p> <p><math>00_B</math> <b>OFF</b>, No Spectrum Extension  <math>01_B</math> <b>RANGE</b>, Range Spectrum Extension  <math>10_B</math> <b>VELOCITY</b>, Velocity Spectrum Extension  <math>11_B</math> <b>Reserved</b>, Reserved</p>
<b>CFAR_CAE</b>	27	rw	<p><b>CFAR CA Engine Enable</b></p> <p>Select whether the CFAR CA Engine is enabled. If set, the output of the CA engine will be written to memory</p> <p><math>0_B</math> <b>OFF</b>, CA Engine Off  <math>1_B</math> <b>ON</b>, CA Engine On</p>
<b>CFAR_GOSE</b>	28	rw	<p><b>CFAR GOS Engine Enable</b></p> <p>Select whether the CFAR GOS Engine is enabled. If set, the output from the GOS engine will be written to memory</p> <p><math>0_B</math> <b>OFF</b>, GOS Engine Off  <math>1_B</math> <b>ON</b>, GOS Engine On</p>

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
CFAREN	31:29	rw	<p><b>CFAR Module Enable</b></p> <p>If set, the CFAR will be used to locate points of interest. Once points of interest have been identified, the results can be filtered using this information and only these points of interest will be written to Radar Memory. The CFAR results can be written to Radar Memory as part of the dataset.</p> <p><math>000_B</math> <b>OFF, OFF</b> CFAR Engines are disabled</p> <p><math>001_B</math> <b>CFARI</b>, Inline CFAR CFAR will analyse the first FFT processed. The results will be used to filter all FFTs from the current measurement cycle.</p> <p><math>010_B</math> <b>CFARO</b>, Offline CFAR The CFAR Unit will analyse all FFTs and the results will be written to the Radar Memory using the CFARCTRL.BASE field as the starting address</p> <p><math>011_B</math> <b>LCLMAXI</b>, Inline Local Maximum The Threshold Unit will analyse the first FFT processed. The results will be used to filter all FFTs from the current measurement cycle</p> <p><math>100_B</math> <b>LCLMAXO</b>, Offline Local Maximum The Threshold Unit will analyse all FFTs and the results will be written to the Radar Memory using the CFARCTRL.BASE field as the starting address</p> <p><math>101_B</math> <b>CFARN</b>, CFAR on NCI Output The CFAR Unit will analyse all the integrated data from the FFTs and the results will be written to the Radar Memory using the CFARCTRL.BASE field as the starting address. In this mode, the Vector Add Unit output is used as the CFAR input so NCICTRL.EN must be enabled for this mode to function correctly. This assumes that the buffer memory contains one FFT result from each attached antenna.</p> <p><math>110_B</math> <b>LCLMAXN</b>, Threshold Comparison based on non-Coherent Integration Output The Local Max Unit will analyse all the integrated data from the FFTs and the results will be written to the Radar Memory using the CFARCTRL.BASE field as the starting address. In this mode, the Vector Add Unit output is used as the Local Max input so NCICTRL.EN must be enabled for this mode to function correctly and the Local Max unit will be comparing the power value output from the integration calculation. This assumes that the buffer memory contains one FFT result from each attached antenna.</p> <p><math>111_B</math> <b>Reserved</b>, Do Not Use</p>

## Signal Processing Unit (SPU)

**Table 681 Access Mode Restrictions of BEx\_CFARCTRL (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE, CFAREN, CFAR_CAE, CFAR_GOSE, EXTNSN, SEWIN	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BASE, CFAREN, CFAR_CAE, CFAR_GOSE, EXTNSN, SEWIN	Default Access Mode (Bus Error on Write)

**Table 682 Reset Values of BEx\_CFARCTRL (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Sideband Control

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

This register configures and enables the statistical information sideband channel.

#### BEx\_SBCTRL (x=0-1)

##### Sideband Control

(0009C<sub>H</sub>+x\*40<sub>H</sub>)

Reset Value: [Table 684](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN</b>	<b>RES</b>										<b>BASE</b>	rw			
rw							r								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>BASE</b>															rw

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> The base address to be used when writing data to Radar Memory. This is a word (256 bit) address relative to the Radar Memory base address. All writes will be 32 byte words. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing a measurement cycle, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.
<b>RES</b>	30:19	r	<b>Reserved</b>

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
EN	31	rw	<b>Enable</b> Enable to Power Analysis Sideband Channel.

**Table 683 Access Mode Restrictions of BEx\_SBCTRL (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BASE, EN	
Otherwise (default)	r	BASE, EN, RES	Default Access Mode (Bus Error on Write)

**Table 684 Reset Values of BEx\_SBCTRL (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Bin Rejection Mask

Every bit in these registers corresponds to an FFT sample point. If the bit is cleared, then the corresponding sample point will either be removed from the data written to EMEM or set to a zero value

This register contains part of the bin rejection mask. Each bit corresponds to a bin of the FFT output. If the bit is cleared, then the corresponding bin will be filtered from the output.

#### BINm\_REJ (m=0-63)

#### Bin Rejection Mask

(000E0<sub>H</sub>+m\*4)

Reset Value: [Table 686](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>B_R31</b>	<b>B_R30</b>	<b>B_R29</b>	<b>B_R28</b>	<b>B_R27</b>	<b>B_R26</b>	<b>B_R25</b>	<b>B_R24</b>	<b>B_R23</b>	<b>B_R22</b>	<b>B_R21</b>	<b>B_R20</b>	<b>B_R19</b>	<b>B_R18</b>	<b>B_R17</b>	<b>B_R16</b>
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>B_R15</b>	<b>B_R14</b>	<b>B_R13</b>	<b>B_R12</b>	<b>B_R11</b>	<b>B_R10</b>	<b>B_R9</b>	<b>B_R8</b>	<b>B_R7</b>	<b>B_R6</b>	<b>B_R5</b>	<b>B_R4</b>	<b>B_R3</b>	<b>B_R2</b>	<b>B_R1</b>	<b>B_R0</b>
rw															

Field	Bits	Type	Description
<b>B_Rn (n=0-31)</b>	n	rw	<b>BIN</b> Set to "1" to allow the corresponding bin through the unit

## Signal Processing Unit (SPU)

Table 685 Access Mode Restrictions of **BINm\_REJ (m=0-63)** sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	B_Rn (n=0-31)	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	B_Rn (n=0-31)	Default Access Mode (Bus Error on Write)

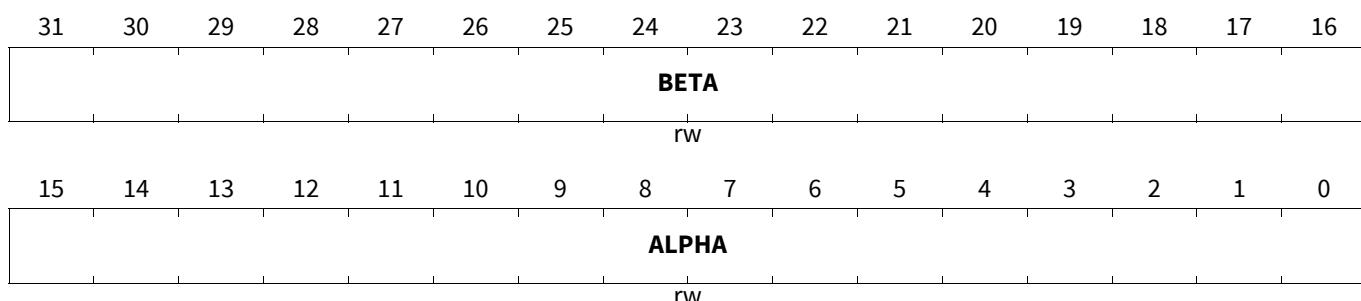
Table 686 Reset Values of **BINm\_REJ (m=0-63)**

Reset Type	Reset Value	Note
Application Reset	FFFF FFFF <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	FFFF FFFF <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Magnitude Approximation Constants

## MAGAPPROX

## Magnitude Approximation Constants

(001E0<sub>H</sub>)Reset Value: [Table 688](#)

Field	Bits	Type	Description
<b>ALPHA</b>	15:0	rw	<b>Alpha Constant</b> Alpha value for Magnitude Approximation Algorithm
<b>BETA</b>	31:16	rw	<b>Beta Constant</b> Beta Value for Magnitude Approximation Algorithm

Table 687 Access Mode Restrictions of **MAGAPPROX** sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ALPHA, BETA	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ALPHA, BETA	Default Access Mode (Bus Error on Write)

## Signal Processing Unit (SPU)

Table 688 Reset Values of MAGAPPROX

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## NCI Antennae Scaling Factor

Each antenna in the system can be individually scaled before integration. The scaling factor is a 16bit unsigned value which is used to adjust the power calculated from each FFT output value. The value is scaled so that the maximum value is equivalent to multiplying by one (approx.).

## NCISCALAR0

NCI Antennae Scaling Factor (001E4<sub>H</sub>) Reset Value: [Table 690](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ANT1															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ANT0															
rw															

Field	Bits	Type	Description
ANT0	15:0	rw	Scaling Factor for Antenna 0
ANT1	31:16	rw	Scaling Factor for Antenna 1

Table 689 Access Mode Restrictions of NCISCALAR0 sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ANT0, ANT1	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ANT0, ANT1	Default Access Mode (Bus Error on Write)

Table 690 Reset Values of NCISCALAR0

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit (SPU)

### NCI Antennae Scaling Factor

Each antenna in the system can be individually scaled before integration. The scaling factor is a 16bit unsigned value which is used to adjust the power calculated from each FFT output value. The value is scaled so that the maximum value is equivalent to multiplying by one (approx.).

### NCISCALAR1

**NCI Antennae Scaling Factor** **(001E8<sub>H</sub>)** **Reset Value: Table 692**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ANT3															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ANT2															
rw															

Field	Bits	Type	Description
ANT2	15:0	rw	Scaling Factor for Antenna 2
ANT3	31:16	rw	Scaling Factor for Antenna 3

**Table 691 Access Mode Restrictions of NCISCALAR1 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ANT2, ANT3	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ANT2, ANT3	Default Access Mode (Bus Error on Write)

**Table 692 Reset Values of NCISCALAR1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### NCI Antennae Scaling Factor

Each antenna in the system can be individually scaled before integration. The scaling factor is a 16bit unsigned value which is used to adjust the power calculated from each FFT output value. The value is scaled so that the maximum value is equivalent to multiplying by one (approx.).

**Signal Processing Unit (SPU)****NCISCALAR2****NCI Antennae Scaling Factor****(001EC<sub>H</sub>)****Reset Value: Table 694**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ANT5															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ANT4															
rw															

Field	Bits	Type	Description
ANT4	15:0	rw	Scaling Factor for Antenna 4
ANT5	31:16	rw	Scaling Factor for Antenna 5

**Table 693 Access Mode Restrictions of NCISCALAR2 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ANT4, ANT5	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ANT4, ANT5	Default Access Mode (Bus Error on Write)

**Table 694 Reset Values of NCISCALAR2**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**NCI Antennae Scaling Factor**

Each antenna in the system can be individually scaled before integration. The scaling factor is a 16bit unsigned value which is used to adjust the power calculated from each FFT output value. The value is scaled so that the maximum value is equivalent to multiplying by one (approx.).

**NCISCALAR3****NCI Antennae Scaling Factor****(001F0<sub>H</sub>)****Reset Value: Table 696**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ANT7															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ANT6															
rw															

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>ANT6</b>	15:0	rw	<b>Scaling Factor for Antenna 6</b>
<b>ANT7</b>	31:16	rw	<b>Scaling Factor for Antenna 7</b>

**Table 695 Access Mode Restrictions of NCISCALAR3 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ANT6, ANT7	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ANT6, ANT7	Default Access Mode (Bus Error on Write)

**Table 696 Reset Values of NCISCALAR3**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### CFAR Configuration

Configuration Parameters for the integrated CFAR module are stored in this register

#### CFARCFG

**CFAR Configuration (001F4<sub>H</sub>) Reset Value: Table 698**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>CABETA</b>															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>CFARSEL</b>	<b>RES</b>	<b>CAWINCELL</b>			<b>CAGUARD</b>						<b>GOSALGO</b>	<b>CAALGO</b>			
rw	r	rw			rw						rw	rw			

Field	Bits	Type	Description
<b>CAALGO</b>	1:0	rw	<b>CA-CFAR Algorithm Select</b> Select the CA-CFAR algorithm 00 <sub>B</sub> <b>CASHCFAR</b> , CASH-CFAR CASH-CFAR algorithm 01 <sub>B</sub> <b>CACFAR</b> , CA-CFAR 10 <sub>B</sub> <b>CAGOCFAR</b> , CAGO-CFAR 11 <sub>B</sub> <b>CASOCFAR</b> , CASO-CFAR

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>GOSALGO</b>	3:2	rw	<b>GOS-CFAR Algorithm Select</b> Select the CA-CFAR algorithm 00 <sub>B</sub> <b>GOSCA</b> , GOSCA-CFAR GOSCA-CFAR Algorithm 01 <sub>B</sub> <b>GOSGOCFAR</b> , GOSGO-CFAR GOSGO-CFAR Algorithm 10 <sub>B</sub> <b>GOSSOCFAR</b> , GOSSO-CFAR GOSSO-CFAR Algorithm 11 <sub>B</sub> <b>GOSSOCFAR</b> , GOSSO-CFAR GOSSO-CFAR Algorithm
<b>CAGUARD</b>	9:4	rw	<b>Guard Cells</b> Number of guard cells in CA-CFAR leading and lagging the cell under test
<b>CAWINCELL</b>	12:10	rw	<b>Window Cells Exponent</b> Exponent of 2 for defining the number of active cells in leading/lagging windows to be averaged in CA-CFAR. 2^(WINCELL) should be less than or equal to 32.
<b>RES</b>	13	r	<b>Reserved</b>
<b>CFARSEL</b>	15:14	rw	<b>Inline CFAR Engine</b> Defines the engine to be used for inline CFAR mode (CFARCTRL.CFAREN=CFARI) if both engines are enabled using the CFARCTRL.CFAR_CAE and CFARCTRL.CFAR_GOSE bits. If the outputs from both engines are not required by the application and only one of the two bits in the CFARCTRL register is set, then this field can be left at AUTO. 00 <sub>B</sub> <b>CA</b> , Use CA Engine 01 <sub>B</sub> <b>GOS</b> , Use GOS Engine 10 <sub>B</sub> <b>BOTH</b> , Use CA Engine on Double Pass, pass1. Use GOS Engine on Double Pass, pass 2 11 <sub>B</sub> <b>AUTO</b> , AUTO Use the engine enabled in the CFARCTRL register. For this to work, only one of the CFAR_CAE and CFAR_GOSE bits can be set for a processing pass
<b>CABETA</b>	31:16	rw	<b>CA-CFAR Beta</b> Additive constant scaling the CA-CFAR threshold

**Table 697 Access Mode Restrictions of CFARCFG sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	CAALGO, CABETA, CAGUARD, CAWINCELL, CFARSEL, GOSALGO	
Otherwise (default)	r	CAALGO, CABETA, CAGUARD, CAWINCELL, CFARSEL, GOSALGO, RES	Default Access Mode (Bus Error on Write)

## Signal Processing Unit (SPU)

Table 698 Reset Values of CFARCFG

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## CFAR Configuration 2

Configuration Parameters for the integrated CFAR module are stored in this register

## CFARCFG2

## CFAR Configuration 2

(001F8<sub>H</sub>)

Reset Value: Table 700

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES					CASHWIN				GOSWINCELL						
				r				rw					rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDXLG					IDXLD				GOSGUARD						
					rw			rw					rw		

Field	Bits	Type	Description
<b>GOSGUARD</b>	5:0	rw	<b>Guard Cells</b> Number of guard cells in GOS-CFAR leading and lagging the cell under test
<b>IDXLD</b>	10:6	rw	<b>Index Lead</b> Index of sorted statistic in leading window in GOS-CFAR
<b>IDXLG</b>	15:11	rw	<b>Index Lag</b> Index of sorted statistic in lagging window in GOS-CFAR
<b>GOSWINCELL</b>	21:16	rw	<b>Window Cells Exponent</b> Value defining the number of active cells in leading/lagging windows to be averaged in GOS-CFAR. WINCELL should be less than or equal to 32. This sets a window size of between 1 and 32 cells. A value of 0 or a value greater than 32 should not be programmed. Behaviour in these cases is undefined
<b>CASHWIN</b>	24:22	rw	<b>CASH Subwindow</b> Exponent of 2 for defining the number of active cells in leading/lagging subwindow when using the CASH algorithm in the CA-CFAR engine. 2^(CASHWIN) should be less than or equal to 32.
<b>RES</b>	31:25	r	<b>Reserved</b>

## Signal Processing Unit (SPU)

Table 699 Access Mode Restrictions of CFARCFG2 sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	CASHWIN, GOSGUARD, GOSWINCELL, IDXLD, IDXLG	
Otherwise (default)	r	CASHWIN, GOSGUARD, GOSWINCELL, IDXLD, IDXLG, RES	Default Access Mode (Bus Error on Write)

Table 700 Reset Values of CFARCFG2

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## CFAR Configuration 3

Configuration Parameters for the integrated CFAR module are stored in this register

CFARCFG3																CFAR Configuration 3				(001FC <sub>H</sub> )				Reset Value: <a href="#">Table 702</a>			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	CHAN5OFFST											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	GOSBETA											

Field	Bits	Type	Description
GOSBETA	15:0	rw	<b>GOS-CFAR Beta</b> Additive constant scaling the GOS-CFAR threshold
CHAN5OFFST	31:16	rw	<b>Channel 5 Address Offset</b> Offset Address added to CFARCTRL.BASE to provide base address for writing GOS-CFAR results. This can be set to 0x0000 when only the GOS-CFAR engine is enabled. This field is a word address, where each word is 32 bytes. The generated address will automatically increment after each write. If a partial word remains at the end of processing a measurement cycle, the data will be padded to 32 bytes and flushed to Radar Memory. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.

## Signal Processing Unit (SPU)

**Table 701 Access Mode Restrictions of CFARCFG3 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	CHAN5OFFST, GOSBETA	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	CHAN5OFFST, GOSBETA	Default Access Mode (Bus Error on Write)

**Table 702 Reset Values of CFARCFG3**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Scalar Addition Operand

Parameter to be used when adjusting the FFT output data

#### SCALARADD

**Scalar Addition Operand** **(00200<sub>H</sub>)** **Reset Value: Table 704**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OPERAND															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OPERAND															
rw															

Field	Bits	Type	Description
<b>OPERAND</b>	31:0	rw	<b>Operand for Scaling</b> Additive Scaling factor for FFT output data. This is 32bit signed integer value which will be added to both the real an complex components of the FFT output data.

**Table 703 Access Mode Restrictions of SCALARADD sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	OPERAND	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	OPERAND	Default Access Mode (Bus Error on Write)

## Signal Processing Unit (SPU)

Table 704 Reset Values of SCALARADD

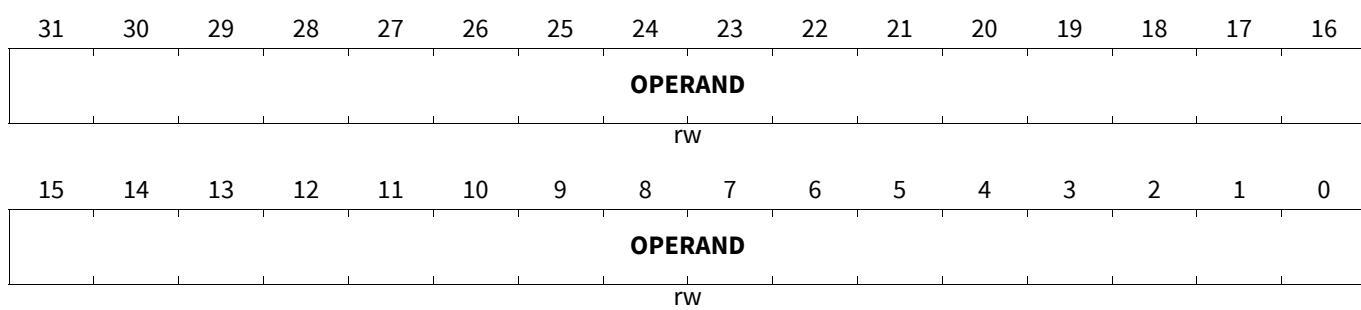
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Scalar Multiplication Operand

Parameter to be used when adjusting the FFT output data

## SCALARMULT

Scalar Multiplication Operand (00204<sub>H</sub>) Reset Value: Table 706



Field	Bits	Type	Description
OPERAND	31:0	rw	<b>Operand for Scaling</b> This is a multiplicative scaling factor to be applied to the FFT output data. It is a 32bit signed integer value with a nominal range of -1 to +1. Both the real and complex components are multiplied by the number.

Table 705 Access Mode Restrictions of SCALARMULT sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	OPERAND	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	OPERAND	Default Access Mode (Bus Error on Write)

Table 706 Reset Values of SCALARMULT

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit (SPU)

### Bin Rejection Unit Control

Control Parameters for the Bin Rejection Unit. The Bin Rejection Unit takes inputs from a 2048 set of registers (one bit per FFT data point) and a 1024 register from the CFAR and Threshold Unit. These bits are used to zero bins or remove them from the outputs data stream. Additionally bins can be compared to a threshold value and set to zero if the threshold is exceeded. The threshold value is set as a magnitude. The Magnitude Approximation of the complex data bins is used for the comparison

#### BINREJCTRL

**Bin Rejection Unit Control** **(00208<sub>H</sub>)** **Reset Value: Table 708**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>RES</b>								<b>VALUE</b>							
r															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>VALUE</b>												<b>LJUST</b>	<b>ZMODE</b>	<b>RMODE</b>	
												rw	rw	rw	

Field	Bits	Type	Description
<b>RMODE</b>	1:0	rw	<b>Bin Rejection Mode</b> 00 <sub>B</sub> <b>OFF</b> , Unit Disabled Pass all data 01 <sub>B</sub> <b>REJ</b> , Bin Rejection remove the selected bins from the output data 10 <sub>B</sub> <b>ZERO</b> , Zero set the selected bins to zero 11 <sub>B</sub> <b>RES</b> , Reserved
<b>ZMODE</b>	2	rw	<b>Threshold Rejection Mode</b> 0 <sub>B</sub> <b>OFF</b> , Unit Disabled Pass all data 1 <sub>B</sub> <b>ZETH</b> , Zero with Threshold Set bins to zero if they exceed the programmed threshold (BINREJCTRL.VALUE)
<b>LJUST</b>	3	rw	<b>Left Justify</b> The 24 bit VALUE bitfield is compared against a 32 bit value. This field controls whether the comparison is made against the 24 MSBs or 24 LSBs 0 <sub>B</sub> <b>RIGHT</b> , Right Justify. Pad with 8 MSBs to create a 32 bit comparison value 1 <sub>B</sub> <b>LEFT</b> , Left Justify. Pad with 8 LSBs to create a 32 bit comparison value
<b>VALUE</b>	27:4	rw	<b>Threshold Value</b> This is the threshold value to be used for comparison. This is a 16 bit magnitude value
<b>RES</b>	31:28	r	<b>Reserved</b>

## Signal Processing Unit (SPU)

**Table 707 Access Mode Restrictions of [BINREJCTRL](#) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	LJUST, RMODE, VALUE, ZMODE	
Otherwise (default)	r	LJUST, RES, RMODE, VALUE, ZMODE	

**Table 708 Reset Values of [BINREJCTRL](#)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Local Maximum Control

Control Parameters for the Local Maximum Unit

#### LCLMAX

**Local Maximum Control** (0020C<sub>H</sub>) Reset Value: [Table 710](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>TSHLD</b>															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>TSHLD</b>								rw							

Field	Bits	Type	Description
<b>WIDTH</b>	1:0	rw	<b>Local Maximum Window Width</b> This bitfield sets the width of the comparison window for the Local Maximum detection. 00 <sub>B</sub> <b>OFF</b> , No Local Maximum 01 <sub>B</sub> <b>THREE</b> , On, with Window Size of Three The bin under consideration will be set to zero if it is a local maximum compared to the adjacent bins in the FFT result 10 <sub>B</sub> <b>FIVE</b> , On, with Window Size of Five The bin under consideration will be set to zero if it is a local maximum compared to the four adjacent bins (index -2 to index +2) in the FFT result 11 <sub>B</sub> <b>RES</b> , Reserved Do Not use

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>TMODE</b>	3:2	rw	<b>Threshold Mode</b> Mode for Bin Magnitude Comparison 00 <sub>B</sub> <b>OFF</b> , No Thresholding 01 <sub>B</sub> <b>UNDER</b> , Reject Bin if Under Threshold 10 <sub>B</sub> <b>OVER</b> , Reject Bin if Over Threshold 11 <sub>B</sub> <b>RES</b> , Reserved
<b>LMODE</b>	5:4	rw	<b>Local Max Mode</b> 00 <sub>B</sub> <b>OFF</b> , Off 01 <sub>B</sub> <b>UNDER</b> , Reject bin if not local maximum 10 <sub>B</sub> <b>OVER</b> , Reject Bin if Local Max 11 <sub>B</sub> <b>RES</b> , Reserved, do not use
<b>CMBN</b>	6	rw	<b>Check Combine</b> This bitfield is used to define how the results of the two possible checks are combined. This field only has an effect if both checks are enabled. 0 <sub>B</sub> <b>OR</b> , OR Check Results Reject bin if either check triggers rejection 1 <sub>B</sub> <b>AND</b> , AND Check Results. Reject bin if both checks trigger a rejection
<b>LJUST</b>	7	rw	<b>Left Justify</b> The 24 bit TSHLD bitfield is compared against a 32 bit value. This field controls whether the comparison is made against the 24 MSBs or 24 LSBs 0 <sub>B</sub> <b>RIGHT</b> , Right Justify. Pad with 8 MSBs to create a 32 bit comparison value 1 <sub>B</sub> <b>LEFT</b> , Left Justify. Pad with 8 LSBs to create a 32 bit comparison value
<b>TSHLD</b>	31:8	rw	<b>Threshold</b> Threshold Value to be Used for Bin Magnitude Comparison

**Table 709 Access Mode Restrictions of LCLMAX sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	CMBN, LJUST, LMODE, TMODE, TSHLD, WIDTH	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	CMBN, LJUST, LMODE, TMODE, TSHLD, WIDTH	Default Access Mode (Bus Error on Write)

**Table 710 Reset Values of LCLMAX**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit (SPU)

### Spare Configuration Register

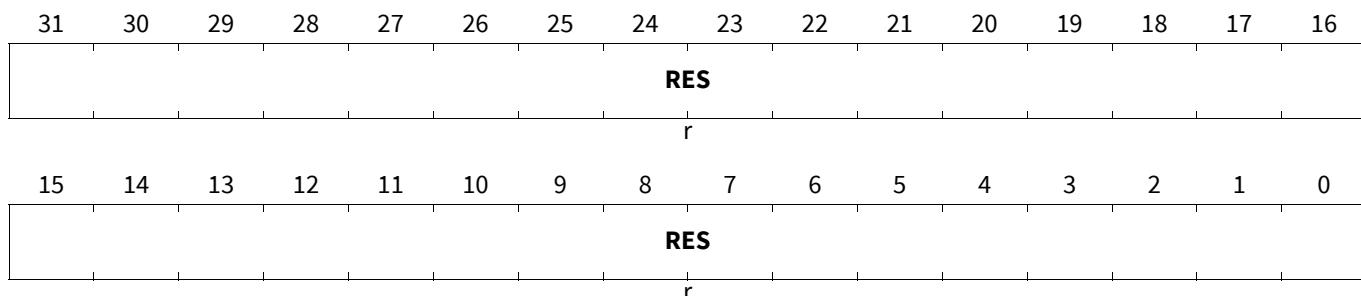
This register is reserved for future expansion of the SPU functionality

#### ACFG2

##### Spare Configuration Register

(00210<sub>H</sub>)

Reset Value: [Table 712](#)



Field	Bits	Type	Description
RES	31:0	r	Reserved

**Table 711 Access Mode Restrictions of ACFG2 sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)

**Table 712 Reset Values of ACFG2**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Register CRC

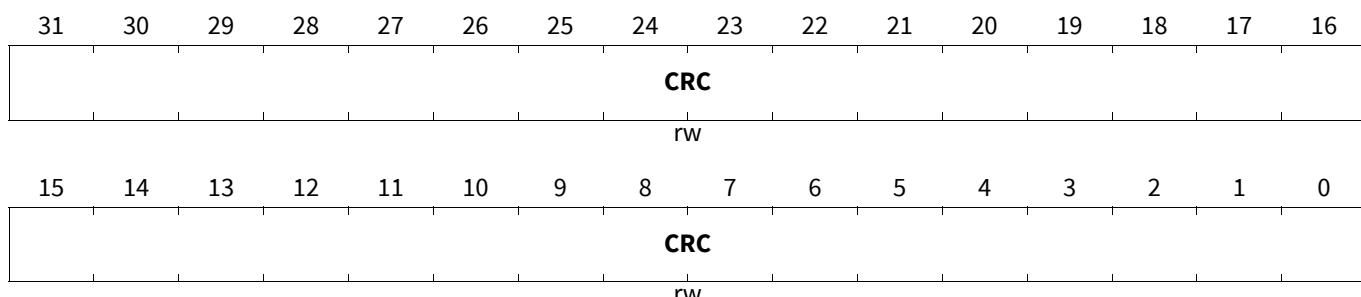
Expected CRC value for all registers from ID\_CONF to CTRL (excluding the CRC register itself)

#### REGCRC

##### Register CRC

(00218<sub>H</sub>)

Reset Value: [Table 714](#)



## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>CRC</b>	31:0	rw	<b>CRC</b> The expected 32 bit CRC of all registers from ID_CONF to CTRL excluding the REGCRC register itself (which is replaced by 0x00000000)

**Table 713 Access Mode Restrictions of REGCRC sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	CRC	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	CRC	Default Access Mode (Bus Error on Write)

**Table 714 Reset Values of REGCRC**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### SPU Control

Control Register for setting the SPU mode and triggering processing operations

#### CTRL

<b>SPU Control</b> (0021C <sub>H</sub> )															Reset Value: <a href="#">Table 716</a>		
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
<b>LAST</b>	<b>RES</b>		<b>XTRIG</b>	<b>ATTN</b>	<b>DIV</b>		<b>NXT_CONF</b>										
rw		r	rw	rw	rw						rw						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
<b>NXT_CONF</b>												<b>MODE</b>	<b>BUSY</b>	<b>TRIG</b>			
												rw	rh	rwh			

Field	Bits	Type	Description
<b>TRIG</b>	0	rwh	<b>SPU Software Trigger</b> Write 1 to trigger the SPU by software. Always reads as 0. 0 <sub>B</sub> <b>NULL</b> , No Operation 1 <sub>B</sub> <b>trigger</b> , Trigger the SPU to commence data processing
<b>BUSY</b>	1	rh	<b>SPU Busy Flag</b> 0 <sub>B</sub> <b>DONE</b> , SPU Idle SPU has finished processing and is guaranteed not to be using EMEM 1 <sub>B</sub> <b>BUSY</b> , SPU Busy SPU is currently processing data

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
MODE	4:2	rw	<p><b>SPU Trigger Mode</b></p> <p>000<sub>B</sub> <b>OFF</b>, SPU is disabled</p> <p>001<sub>B</sub> <b>INT</b>, Internal Trigger</p> <p>010<sub>B</sub> <b>EXT</b>, External Trigger</p> <p>011<sub>B</sub> <b>SPU0</b>, Trigger on SPU0 Done</p> <p>100<sub>B</sub> <b>SPU1</b>, Trigger on SPU1 Done</p> <p>101<sub>B</sub> <b>RELOAD</b>, Reload configuration from memory with no processing</p> <p>110<sub>B</sub> <b>SW</b>, Software Trigger</p> <p>111<sub>B</sub> <b>STOP</b>, SPU will stop at end of current operation.</p>
NXT_CONF	23:5	rw	<p><b>Next Configuration Base Address</b></p> <p>This is the base address in the configuration RAM of the next set of configuration settings for the SPU. This is a byte address offset into the configuration RAM (i.e. first location in the configuration RAM is address 0) but it must be 64 bit aligned (i.e. bits [2:0] of the address must be 0).</p>
DIV	25:24	rw	<p><b>Clock Division Ratio</b></p> <p>Sets the rate at which data is output from the LOADER into the FFT engine (or the UNLOADER if the FFT is bypassed). This allows the processing rate to be lowered to reduce power if either full performance is not required or the MATH2 is too heavily loaded to keep up with the FFT pipeline running at full speed or the histogram function is enabled.</p> <p>00<sub>B</sub> <b>UNITY</b>, 1:1 Clock Ratio Runs the Loader output at the same speed as the main SPU clockfrequency</p> <p>01<sub>B</sub> <b>DIV2</b>, 1:2 Clock Ratio Runs the Loader output at half the speed as the main SPU clockfrequency</p> <p>10<sub>B</sub> <b>DIV4</b>, 1:4 Clock Ratio Runs the Loader output at quarter the speed of the main SPU clockfrequency</p> <p>11<sub>B</sub> <b>DIV8</b>, 1:8 Clock Ratio Runs the Loader output at one-eighth the speed of the main SPU clockfrequency</p>
ATTN	26	rw	<p><b>Generate Service Request Interrupt</b></p> <p>If this bit is set in a loaded configuration and the related mask bit, STAT.INTMSK(0), is also set, then a service request interrupt will be generated when the execution of the configuration is completed, even if this is not the last configuration in a linked list.</p> <p>0<sub>B</sub> <b>OFF</b>, No effect</p> <p>1<sub>B</sub> <b>ON</b>, A service request interrupt will be generated when execution of the loaded configuration has completed</p>
XTRIG	27	rw	<p><b>Cross Trigger</b></p> <p>If this bit is set a "DONE" event will be triggered at the end of execution of the current configuration (specifically when the last data has been written to Radar memory). This will be used by the SPU0 or SPU1 trigger modes defined for the CTRL.MODE bitfield</p>
RES	30:28	r	<b>Reserved</b>

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
LAST	31	rw	<b>Last Configuration</b> If this bit is set while loading a configuration, this will be the last configuration processed and no further configurations will be loaded

**Table 715 Access Mode Restrictions of CTRL sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rh	BUSY	
	rw	ATTN, DIV, LAST, MODE, NXT_CONF, XTRIG	
	wh	TRIG	
Otherwise (default)	r	ATTN, DIV, LAST, MODE, NXT_CONF, RES, XTRIG	Default Access Mode (Bus Error on Write)
	rh	BUSY, TRIG	

**Table 716 Reset Values of CTRL**

Reset Type	Reset Value	Note
Application Reset	0200 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0200 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Dataset Metadata

This register is duplicated so that the stored information is available for each set of configuration options used in "Double Pass" mode.

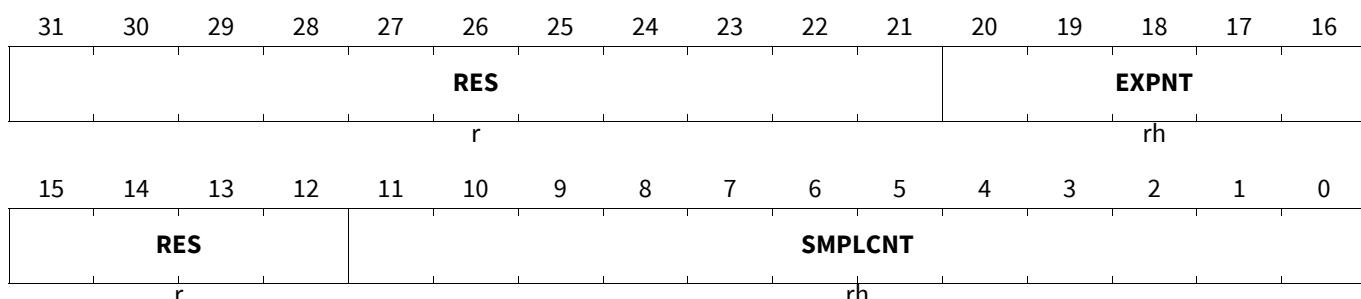
Secondary Information describing the data that has just been processed

#### MDq\_METADATA (q=0-1)

##### Dataset Metadata

(00220<sub>H</sub>+q\*88<sub>H</sub>)

Reset Value: [Table 718](#)



## Signal Processing Unit (SPU)

Field	Bits	Type	Description
SMPLCNT	11:0	rh	<b>Sample Count</b> Number of samples per FFT written to memory. This is useful when using the CFAR inline as the number of bins in the FFT result that will pass the threshold test is not known ahead of processing. The field contains the actual number of samples written to the Radar Memory
RES	15:12, 31:21	r	<b>Reserved</b>
EXPNT	20:16	rh	<b>Common Exponent</b> This indicates the optimum alignment correction that could have been applied if reformatting to 16 bit precision. If "n" is the minimum number of redundant sign bits present in the data at the FFT output, it will be set to $(16_D - n)$ . This can then be used as the value of UNLDR_CONF.EXPNT to be programmed for the next measurement cycle if it is intended to use 16 bit precision data. The maximum permissible value for this field is $16_D$

Table 717 Access Mode Restrictions of MDq\_METADATA (q=0-1) sorted by descending priority

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
	rh	EXPNT, SMPLCNT	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	EXPNT, SMPLCNT	

Table 718 Reset Values of MDq\_METADATA (q=0-1)

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Bin Rejection Unit Tracking**

This register is duplicated so that the stored information is available for each set of configuration options used in "Double Pass" mode.

This register provides information about the bin rejection unit behaviour for the data that has just been processed. The value read is dynamically updated while the SPU is processing data and is only stable when the SPU has completed processing and has written all data to memory.

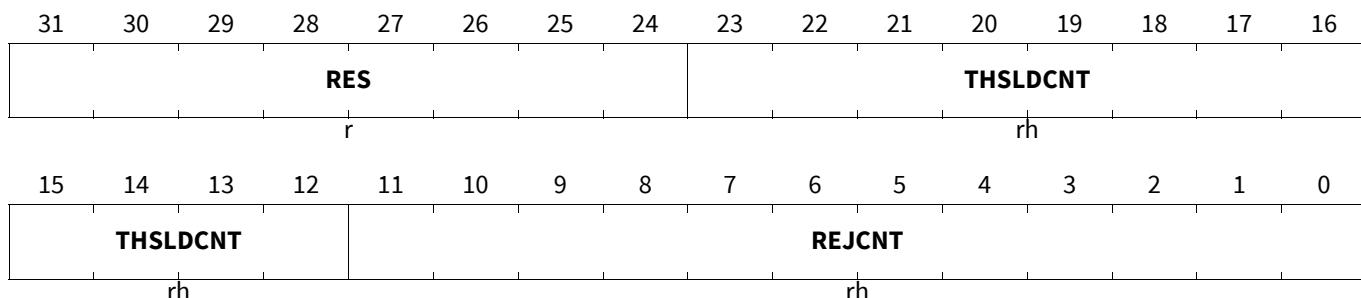
## Signal Processing Unit (SPU)

### MDq\_BINCOUNT (q=0-1)

Bin Rejection Unit Tracking

(00224<sub>H</sub>+q\*88<sub>H</sub>)

Reset Value: [Table 720](#)



Field	Bits	Type	Description
REJCNT	11:0	rh	<b>Rejection Count</b> Number of samples rejected or set to zero by the bin rejection unit based on the inputs from the CFAR or Local Max functions combined with the static mask registers, BINm_REJ. Always contains the value for the last FFT processed
THSLDCNT	23:12	rh	<b>Threshold Function Count</b> Number of samples set to zero by the bin rejection unit threshold function. Always contains the value for the last FFT processed
RES	31:24	r	<b>Reserved</b>

**Table 719 Access Mode Restrictions of MDq\_BINCOUNT (q=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rh	REJCNT, THSLDCNT	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	REJCNT, THSLDCNT	

**Table 720 Reset Values of MDq\_BINCOUNT (q=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Bin Acceptance Mask

Every bit in these registers corresponds to an FFT sample point. If the bit is cleared, then Bin Rejection Unit has either removed the corresponding sample point from the data written to EMEM or set it to a zero value. This register set supports a maximum FFT result size of 1024 sample points and excludes data on points set to zero by the bin rejection unit threshold function. The data contained in the register set is guaranteed to be valid only when the SPU is idle and is relevant for the last FFT processed.

## Signal Processing Unit (SPU)

Each register contains part of the bin acceptance mask. Each bit corresponds to a bin of the FFT output. If the bit is cleared, then the corresponding bin has been filtered from the output or set to zero by either the Local Max Unit, the CFAR Unit or the static mask configuration.

### MDq\_MASKm\_ACCEPT (m=0-31;q=0-1)

Bin Acceptance Mask

( $00228_H + q*88_H + m*4$ )

Reset Value: [Table 722](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>B_A31</b>	<b>B_A30</b>	<b>B_A29</b>	<b>B_A28</b>	<b>B_A27</b>	<b>B_A26</b>	<b>B_A25</b>	<b>B_A24</b>	<b>B_A23</b>	<b>B_A22</b>	<b>B_A21</b>	<b>B_A20</b>	<b>B_A19</b>	<b>B_A18</b>	<b>B_A17</b>	<b>B_A16</b>
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>B_A15</b>	<b>B_A14</b>	<b>B_A13</b>	<b>B_A12</b>	<b>B_A11</b>	<b>B_A10</b>	<b>B_A9</b>	<b>B_A8</b>	<b>B_A7</b>	<b>B_A6</b>	<b>B_A5</b>	<b>B_A4</b>	<b>B_A3</b>	<b>B_A2</b>	<b>B_A1</b>	<b>B_A0</b>
rh															

Field	Bits	Type	Description
<b>B_An (n=0-31)</b>	n	rh	<b>BIN</b> Set to "1" if the corresponding bin has passed through the unit without either being rejected or set to zero based on input from the CFAR or Local Max unit

**Table 721 Access Mode Restrictions of MDq\_MASKm\_ACCEPT (m=0-31;q=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rh	B_An (n=0-31)	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	rh	B_An (n=0-31)	Default Access Mode (Bus Error on Write)

**Table 722 Reset Values of MDq\_MASKm\_ACCEPT (m=0-31;q=0-1)**

Reset Type	Reset Value	Note
Application Reset	FFFF FFFF <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	FFFF FFFF <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Input DMA Count

Number of Read Transactions addressed to Radar Memory since last reset

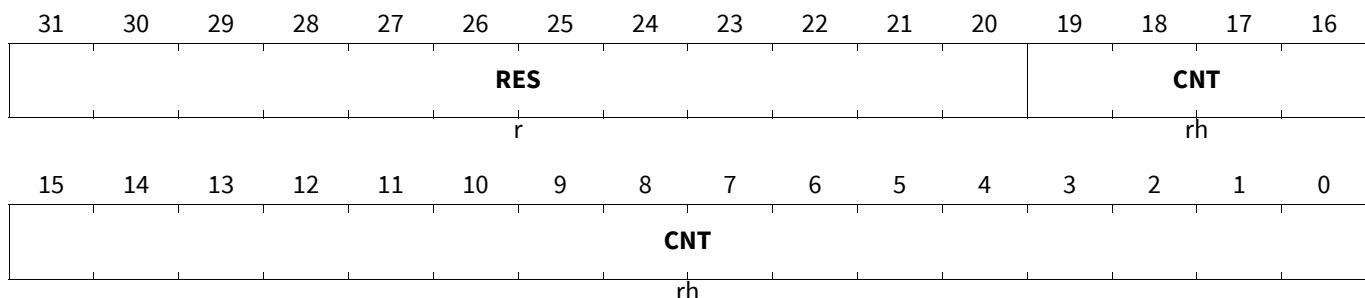
## Signal Processing Unit (SPU)

### IDMCNT

#### Input DMA Count

(00330<sub>H</sub>)

Reset Value: [Table 724](#)



Field	Bits	Type	Description
CNT	19:0	rh	<b>Access Count</b> Counter incremented every time The Input DMA reads from data memory. Write 01b to CNTCLR.CLR to clear
RES	31:20	r	<b>Reserved</b>

**Table 723 Access Mode Restrictions of IDMCNT sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
	rh	CNT	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	CNT	

**Table 724 Reset Values of IDMCNT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Input Buffer Memory Count

Number of Write Transactions addressed to Input Buffer Memory since last reset

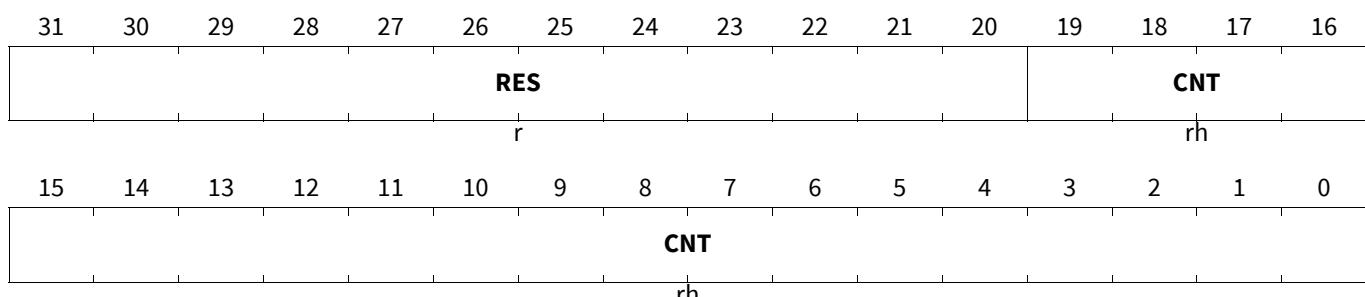
## Signal Processing Unit (SPU)

### IBMCNT

#### Input Buffer Memory Count

(00334<sub>H</sub>)

Reset Value: [Table 726](#)



Field	Bits	Type	Description
CNT	19:0	rh	<b>Access Count</b> Counter incremented every time a transaction occurs. Write 01b to CNTCLR.CLR to clear
RES	31:20	r	<b>Reserved</b>

**Table 725 Access Mode Restrictions of IBMCNT sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
	rh	CNT	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	CNT	

**Table 726 Reset Values of IBMCNT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Input Buffer Memory Read Count

Number of Read Transactions addressed to Input Buffer Memory since last reset

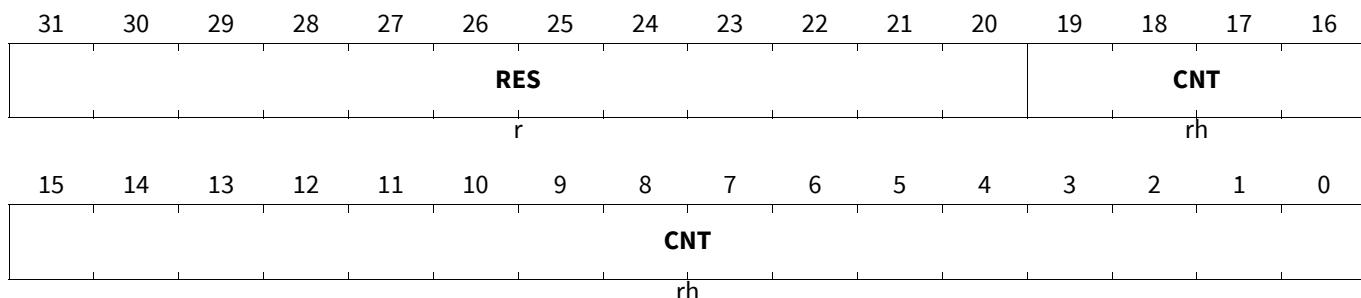
## Signal Processing Unit (SPU)

### LDRCNT

#### Input Buffer Memory Read Count

(00338<sub>H</sub>)

Reset Value: [Table 728](#)



Field	Bits	Type	Description
CNT	19:0	rh	<b>Access Count</b> Counter incremented every time a transaction occurs. Write 01b to CNTCLR.CLR to clear
RES	31:20	r	<b>Reserved</b>

**Table 727 Access Mode Restrictions of LDRCNT sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
	rh	CNT	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	CNT	

**Table 728 Reset Values of LDRCNT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

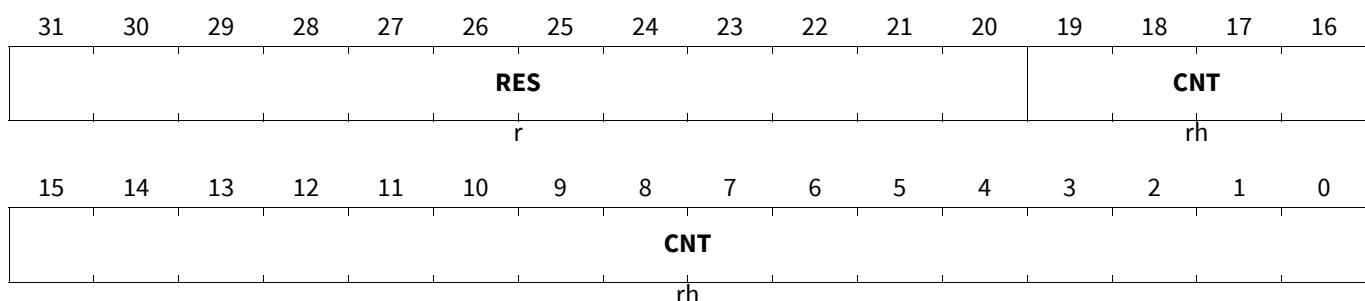
### FFT Load Count

Number of Data Elements pushed into the FFT accelerator since last reset

## Signal Processing Unit (SPU)

### FFTWCNT

#### FFT Load Count

(0033C<sub>H</sub>)Reset Value: [Table 730](#)

Field	Bits	Type	Description
CNT	19:0	rh	<b>Access Count</b> Counter incremented every time a transaction occurs. Write 01b to CNTCLR.CLR to clear
RES	31:20	r	<b>Reserved</b>

**Table 729 Access Mode Restrictions of FFTWCNT sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
	rh	CNT	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	CNT	

**Table 730 Reset Values of FFTWCNT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### FFT Unload Count

Number of Data Elements passed to the Unloader Module since last reset. This counter will normally count the number of data words unloaded from the FFT module but will still increment even if the FFT module is bypassed.

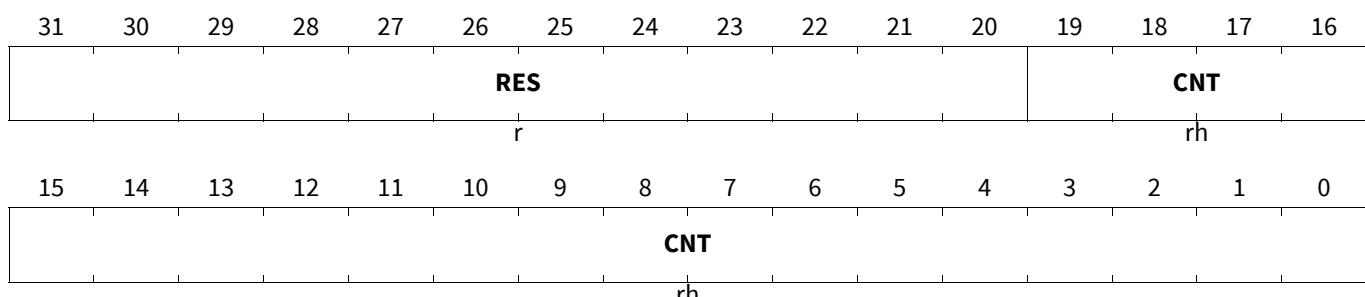
## Signal Processing Unit (SPU)

### FFTRCNT

#### FFT Unload Count

(00340<sub>H</sub>)

Reset Value: [Table 732](#)



Field	Bits	Type	Description
CNT	19:0	rh	<b>Access Count</b> Counter incremented every time a transaction occurs. Write 01b to CNTCLR.CLR to clear
RES	31:20	r	<b>Reserved</b>

**Table 731 Access Mode Restrictions of FFTRCNT sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
	rh	CNT	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	CNT	

**Table 732 Reset Values of FFTRCNT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Output Buffer Memory Write Count

Number of Write Transactions addressed to Output Buffer Memory since last reset

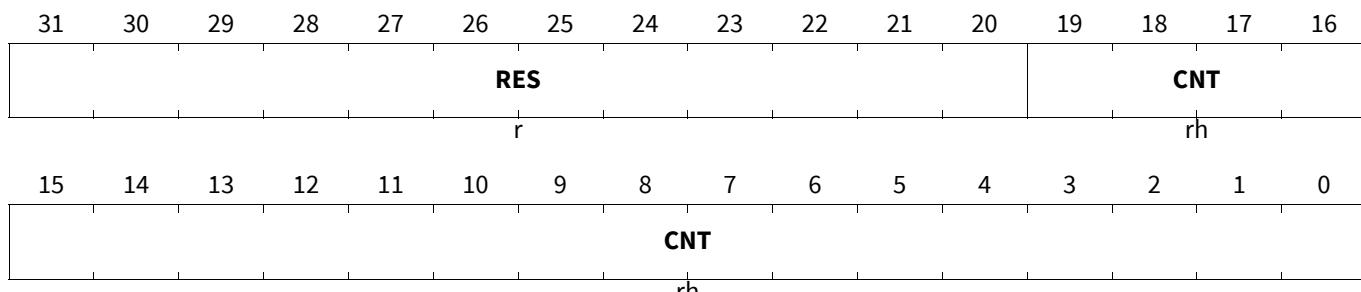
## Signal Processing Unit (SPU)

### ULDRCNT

#### Output Buffer Memory Write Count

(00344<sub>H</sub>)

Reset Value: [Table 734](#)



Field	Bits	Type	Description
CNT	19:0	rh	<b>Access Count</b> Counter incremented every time a transaction occurs. Write 01b to CNTCLR.CLR to clear
RES	31:20	r	<b>Reserved</b>

**Table 733 Access Mode Restrictions of ULDRCNT sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
	rh	CNT	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	CNT	

**Table 734 Reset Values of ULDRCNT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

#### Output Buffer Memory Read Count

Number of Read Transactions addressed to Output Buffer Memory since last reset

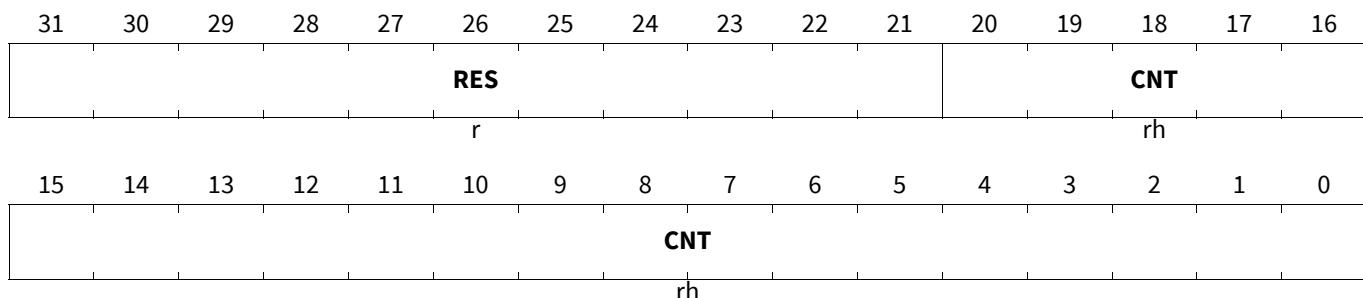
## Signal Processing Unit (SPU)

### ODMCNT

#### Output Buffer Memory Read Count

(00348<sub>H</sub>)

Reset Value: [Table 736](#)



Field	Bits	Type	Description
CNT	20:0	rh	<b>Access Count</b> Counter incremented every time a transaction occurs. Write 01b to CNTCLR.CLR to clear
RES	31:21	r	<b>Reserved</b>

**Table 735 Access Mode Restrictions of ODMCNT sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
	rh	CNT	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	CNT	

**Table 736 Reset Values of ODMCNT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Bin Rejection Unit Load Count

Number of Data Transactions presented at the input of the Bin Rejection Unit since last reset. This counter will be incremented for every data sample regardless of whether or not it is used by the Bin Rejection Unit

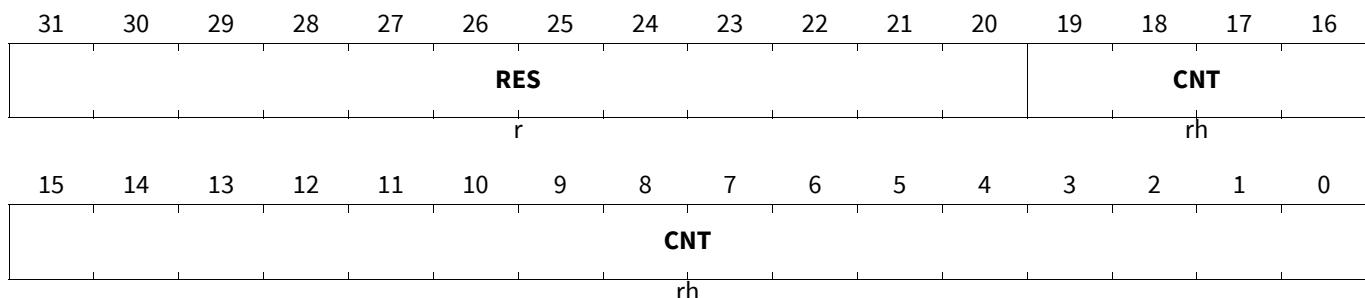
## Signal Processing Unit (SPU)

### BRCNT

#### Bin Rejection Unit Load Count

(0034C<sub>H</sub>)

Reset Value: [Table 738](#)



Field	Bits	Type	Description
CNT	19:0	rh	<b>Access Count</b> Counter incremented every time a transaction occurs. Write 01b to CNTCLR.CLR to clear
RES	31:20	r	<b>Reserved</b>

**Table 737 Access Mode Restrictions of BRCNT sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
	rh	CNT	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	CNT	

**Table 738 Reset Values of BRCNT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### CFAR Unit Load Count

The number of Data Transactions loaded into the CFAR or LMAX Units since last reset. A data transaction is considered as loaded even if it is then discarded as not required because of an over size spectrum extension window.

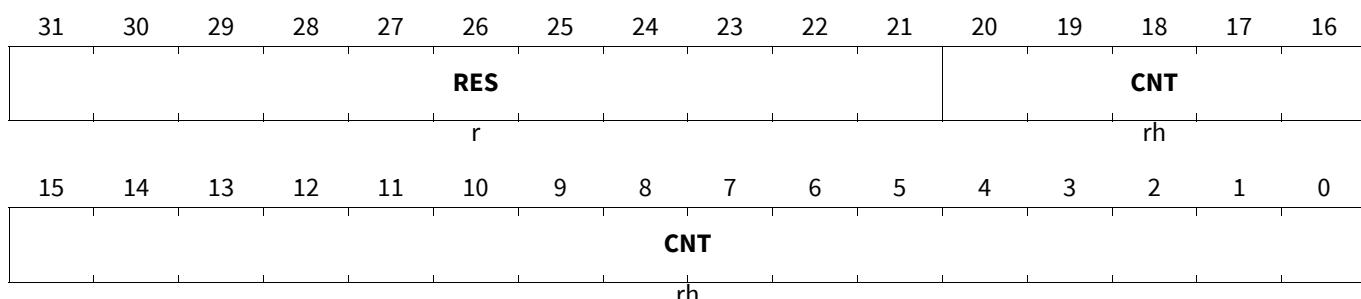
## Signal Processing Unit (SPU)

### CFARCNT

#### CFAR Unit Load Count

(00350<sub>H</sub>)

Reset Value: [Table 740](#)



Field	Bits	Type	Description
CNT	20:0	rh	<b>Access Count</b> Counter incremented every time a data word is clocked into the CFAR module. Write 01b to CNTCLR.CLR to clear
RES	31:21	r	<b>Reserved</b>

**Table 739 Access Mode Restrictions of CFARCNT sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
	rh	CNT	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	CNT	

**Table 740 Reset Values of CFARCNT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Output DMA Port Write Count

The ODMACNT registers count transactions on the ODM write ports. The registers are allocated to ports as follows:

1. Port1 ODMACNT(0) - FFT Data
2. Port2 ODMACNT(1) - Log2 Signal Power
3. Port3 ODMACNT(2) - Stats
4. Port4 ODMACNT(3) - Local Max/CFAR CA
5. Port5 ODMACNT(4) - CFAR GOS
6. Port6 ODMACNT(5) - Vector Add/Sum (NCI)
7. Port7 ODMACNT(6) - sum, sum-abs, sum\_antenna
8. Port8 ODMACNT(7) - Log2 Power Sum.

## Signal Processing Unit (SPU)

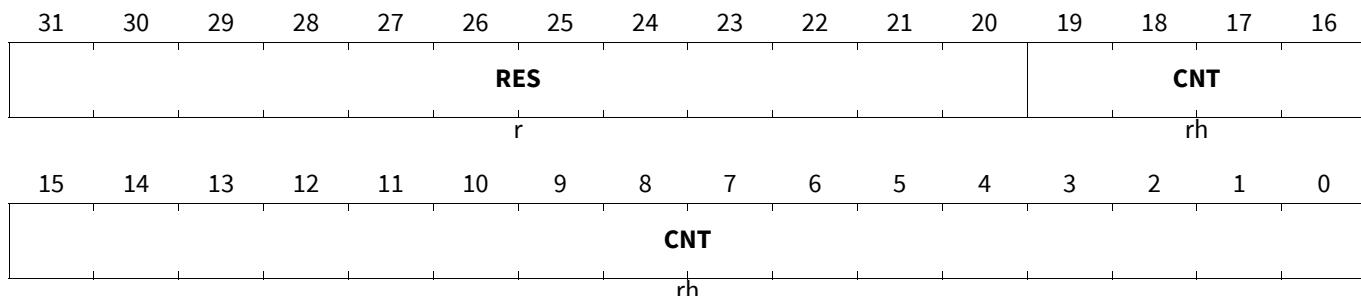
Number of Write Transactions addressed to Output DMA Port since the last reset of the counter

### ODMACNT<sub>p</sub> (p=0-7)

**Output DMA Port Write Count**

(00354<sub>H</sub>+p\*4)

**Reset Value: Table 742**



Field	Bits	Type	Description
CNT	19:0	rh	<b>Access Count</b> Counter incremented every time the Output DMA Port writes to Radar Memory. Write 01b to CNTCLR.CLR to clear
RES	31:20	r	<b>Reserved</b>

**Table 741 Access Mode Restrictions of ODMACNT<sub>p</sub> (p=0-7) sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
	rh	CNT	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	CNT	

**Table 742 Reset Values of ODMACNT<sub>p</sub> (p=0-7)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit (SPU)

### Safety Counter Clear

#### CNTCLR

##### Safety Counter Clear

(00374<sub>H</sub>)

Reset Value: [Table 744](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES										SELECT			CNTTST		
r										rw				rwh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CLR			rwh		

Field	Bits	Type	Description
CLR	1:0	rwh	<b>Clear</b> Clear the counters tracking memory transactions. Always reads as b00 00 <sub>B</sub> <b>NOP</b> , No Operation. Invalid value. SMSTSAT.SMCTRLSTS will be set 01 <sub>B</sub> <b>NOP1</b> , No Operation 10 <sub>B</sub> <b>CLR</b> , Clear All Counters 11 <sub>B</sub> <b>NOP3</b> , No Operation. Invalid value. SMSTSAT.SMCTRLSTS will be set
RES	15:2, 31:23	r	<b>Reserved</b>
CNTTST	17:16	rwh	<b>Monitor Counter Test</b> Inject a deliberate error into one of the counter values to test functionality of the monitoring software. The error will be injected into the next read of the selected register. The bitfield will read as 01 <sub>B</sub> once the error has been injected. 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMSTAT.SMCTRLSTS set 01 <sub>B</sub> <b>OFF</b> , No Error Injected 10 <sub>B</sub> <b>ON</b> , Error Injected 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMSTAT.SMCTRLSTS set
SELECT	22:18	rw	<b>Counter Select</b> Address field used to select the Monitor Counter being tested. 0 <sub>D</sub> selects IDMCNT, 16 <sub>D</sub> selects ODMCNT7. Values in between select the intervening registers in address order.

**Table 743 Access Mode Restrictions of CNTCLR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	SELECT	
	rwh	CLR, CNTTST	
Otherwise (default)	r	RES, SELECT	Default Access Mode (Bus Error on Write)
	rh	CLR, CNTTST	

## Signal Processing Unit (SPU)

Table 744 Reset Values of CNTCLR

Reset Type	Reset Value	Note
Application Reset	0001 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0001 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## SPU Monitor

MONITOR  
SPU Monitor (00378<sub>H</sub>) Reset Value: Table 746

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>RES</b>	<b>ODM_BUSY</b>	<b>M2_B_USY</b>	<b>UL_BU SY</b>	<b>M1_B_USY</b>	<b>LDR_B_USY</b>	<b>IDM_B_USY</b>	<b>RES</b>								<b>SAMPLE</b>
r	rh	rh	rh	rh	rh	rh	rh	r							rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				<b>SAMPLE</b>	<b>RES</b>										<b>RAMP</b>
				rh	r										rh

Field	Bits	Type	Description
<b>RAMP</b>	10:0	rh	<b>Ramp Counter</b> Ramp Count for current measurement cycle. Only active when data is being loaded from the RIF
<b>RES</b>	11, 23, 31:30	r	<b>Reserved</b>
<b>SAMPLE</b>	22:12	rh	<b>Sample Count</b> Sample count for current ramp. Only active when data is being loaded from the RIF
<b>IDM_BUSY</b>	24	rh	<b>IDM Busy</b>
<b>LDR_BUSY</b>	25	rh	<b>Loader Busy</b>
<b>M1_BUSY</b>	26	rh	<b>MATH1 Unit Busy</b>
<b>UL_BUSY</b>	27	rh	<b>Unloader Busy</b>
<b>M2_BUSY</b>	28	rh	<b>MATH2 Busy</b>
<b>ODM_BUSY</b>	29	rh	<b>ODM Busy</b>

## Signal Processing Unit (SPU)

**Table 745 Access Mode Restrictions of MONITOR sorted by descending priority**

Mode Name	Access Mode		Description
Otherwise and Write Accesses	r	RES	Read Only
	rh	IDM_BUSY, LDR_BUSY, M1_BUSY, M2_BUSY, ODM_BUSY, RAMP, SAMPLE, UL_BUSY	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	IDM_BUSY, LDR_BUSY, M1_BUSY, M2_BUSY, ODM_BUSY, RAMP, SAMPLE, UL_BUSY	

**Table 746 Reset Values of MONITOR**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Safety Mechanism Control Functions

This register provides access to the control functions for safety mechanisms which are not normally expected to be modified as part of the normal configuration of the SPU.

#### SMCTRL

**Safety Mechanism Control Functions (0037C<sub>H</sub>) Reset Value: Table 748**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				RMTAERR		BPCRC		RIFCRC		REGCRCEN		DATACRCEN		CTRLCRCEN	
r				rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
CTRLCRCEN	1:0	rw	<b>Control CRC Enable</b> This bitfield allows control of the CRC blocks monitoring data invariant signals. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit. 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>OFF</b> , CRC Monitors are disabled 10 <sub>B</sub> <b>ON</b> , CRC Monitors are enabled 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>DATACRCEN</b>	3:2	rw	<p><b>Data CRC Enable</b></p> <p>This bitfield allows control of the CRC blocks monitoring data dependent signals. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit.</p> <p>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set      01<sub>B</sub> <b>OFF</b>, CRC Monitors are disabled      10<sub>B</sub> <b>ON</b>, CRC Monitors are enabled      11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>
<b>REGCRCEN</b>	5:4	rw	<p><b>Register CRC Enable</b></p> <p>Enable a Periodic CRC check of the register values against a known CRC stored in REGCRC.CRC. This checks registers at addresses between ID_CONF and CTRL inclusive but excluding the REGCRC register. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit</p> <p>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set      01<sub>B</sub> <b>OFF</b>, CRC is disabled      10<sub>B</sub> <b>ON</b>, CRC is enabled      11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>
<b>RIFCRC</b>	7:6	rw	<p><b>RIF Data CRC Check Enable</b></p> <p>Enable a CRC Check on the incoming RIF data. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit</p> <p>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set      01<sub>B</sub> <b>OFF</b>, CRC is disabled      10<sub>B</sub> <b>ON</b>, CRC is enabled      11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>
<b>BPCRC</b>	9:8	rw	<p><b>Bypass Data CRC Check Enable</b></p> <p>Enable a CRC Check on the bypassed RIF data. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit</p> <p>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set      01<sub>B</sub> <b>OFF</b>, CRC is disabled      10<sub>B</sub> <b>ON</b>, CRC is enabled      11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>
<b>RMTAERR</b>	11:10	rw	<p><b>Radar Memory Access Address Error Enable</b></p> <p>Enable an alarm to be generated if an access to the Radar Memory returns an error due to an invalid address. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit</p> <p>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set      01<sub>B</sub> <b>OFF</b>, Access Address Error is disabled      10<sub>B</sub> <b>ON</b>, Access Address Error is enabled      11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>
<b>RES</b>	31:12	r	<b>Reserved</b>

## Signal Processing Unit (SPU)

**Table 747 Access Mode Restrictions of SMCTRL sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN and Safety ENDINIT	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BPCRC, CTRLCRCEN, DATACRCEN, REGCRCEN, RIFCRC, RMTAERR	
Otherwise (default)	r	BPCRC, CTRLCRCEN, DATACRCEN, REGCRCEN, RES, RIFCRC, RMTAERR	Default Access Mode (Bus Error on Write)

**Table 748 Reset Values of SMCTRL**

Reset Type	Reset Value	Note
Application Reset	0000 0555 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0555 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Safety Mechanism Status

The SPU contains hardware safety mechanisms which monitor key areas of the SPU for correct functioning. In the event of an alarm being triggered by one of the safety mechanisms, this register can be read to determine which of the safety mechanisms has detected an error

#### SMSTAT

**Safety Mechanism Status** (00380<sub>H</sub>) **Reset Value:** [Table 750](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
														SMCT RLCLR	SMCT RLSTS
														w	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								SMSCLR	RES		SMSTAT				
								w		r			rh		

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>SMSTAT</b>	5:0	rh	<p><b>Safety Mechanism Status</b></p> <p>Each bit will be set to one if the associated safety mechanism has detected a fault. The flags can be cleared by writing <math>1_b</math> to SMSTAT.SMSCLR. The values read from this bitfield are interpreted as follows (multiple failure conditions will cause the individual values to be summed):</p> <ul style="list-style-type: none"> <li><math>01_H</math> <b>RIFCRC</b>, Check of computed CRC for RIF data against received value has failed</li> <li><math>02_H</math> <b>REGCRC</b>, Check of computed Register CRC against register value has failed</li> <li><math>04_H</math> <b>BYPASSCRC</b>, Check of computed Bypass Data CRC against value read from radar memory has failed</li> <li><math>08_H</math> <b>RDECC</b>, ECC Check Fail on Read from Radar Memory</li> <li><math>10_H</math> <b>RMCTRL</b>, Radar Memory Access Control Signals failed consistency check</li> <li><math>20_H</math> <b>RMTAERR</b>, Access to Radar Memory has resulted in an Error</li> </ul>
<b>RES</b>	7:6, 15:9, 31:18	r	<b>Reserved</b>
<b>SMSCLR</b>	8	w	<p><b>Clear Safety Mechanism Status</b></p> <p>Write <math>1_b</math> to this bit to clear the safety mechanism status flags. This bit will always read <math>0_b</math>.</p>
<b>SMCTRLSTS</b>	16	rh	<p><b>Safety Mechanism Control Status</b></p> <p>Safety Mechanism Control Logic Check has failed (inconsistent logic state)</p>
<b>SMCTRLCLR</b>	17	w	<p><b>Clear SMCTRL Status Flag</b></p> <p>Write <math>1_b</math> to this bit to clear the SMCTRL status flag. This bit will always read <math>0_b</math>.</p>

Table 749 Access Mode Restrictions of **SMSTAT** sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rh	SMCTRLSTS, SMSTAT	
	w	SMCTRLCLR, SMSCLR	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rX	SMCTRLCLR, SMSCLR	
	rh	SMCTRLSTS, SMSTAT	

## Signal Processing Unit (SPU)

**Table 750 Reset Values of SMSTAT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Safety Mechanism Control Functions (User)

This register provides access to the control functions for safety mechanisms not used as part of the normal configuration of the SPU. The various test bitfields defined in this register are used to test that the safety mechanisms can generate alarms as intended as a connectivity test only. It is therefore not necessary (or desirable) for the SPU to be running while these tests are run.

#### SMUSER

#### Safety Mechanism Control Functions [User] (00384<sub>H</sub>)

Reset Value: [Table 752](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		RMTAERRTST		RDECCTST		RMCTRLTST		BPCRCTST		RIFCRCTST					
r				rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CINIT					
r															rw

Field	Bits	Type	Description
CINIT	1:0	rw	<b>Monitor CRC Unit Initialise</b> Write 10 <sub>b</sub> to initialise the Monitor CRC Units. The CRC monitors will be held in the initialisation state until 01 <sub>b</sub> is written. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit, 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>NOP</b> , No Effect 10 <sub>B</sub> <b>ON</b> , Monitor CRC Units Initialised 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set
RES	15:2, 31:26	r	<b>Reserved</b>
RIFCRCTST	17:16	rw	<b>Test RIF CRC</b> Inject a deliberate error into the mechanism checking the CRC of RIF data to test the alarm generation. When set to 10 <sub>b</sub> , all RIF CRC checks should fail until this field is written with 01 <sub>b</sub> . Writing an invalid value will set the SMSTAT.SMCTRLSTS bit. 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>OFF</b> , No Error Injected 10 <sub>B</sub> <b>ON</b> , Error Injected 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>BPCRCTST</b>	19:18	rw	<p><b>Test Bypass CRC</b></p> <p>Inject a deliberate error into into the mechanism checking the CRC of bypass the data totest the alarm generation. When set to 10b, all RIF CRC checks should fail until this field iswritten with 01b. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit.</p> <p><math>00_B</math> <b>ERR0</b>, Invalid. SMCTRLSTS will be set  <math>01_B</math> <b>OFF</b>, No Error Injected  <math>10_B</math> <b>ON</b>, Error Injected  <math>11_B</math> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>
<b>RMCTRLTST</b>	21:20	rw	<p><b>Test Radar Memory Control</b></p> <p>Inject a deliberate error into into the mechanism checking the consistency of the RadarMemory control signals to test the alarm generation. When set to 10<sub>b</sub>, an alarm willbe flagged until this field is written with 01<sub>b</sub>. Writing an invalid value will setthe SMSTAT.SMCTRLSTS bit.</p> <p><math>00_B</math> <b>ERR0</b>, Invalid. SMCTRLSTS will be set  <math>01_B</math> <b>OFF</b>, No Error Injected  <math>10_B</math> <b>ON</b>, Error Injected  <math>11_B</math> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>
<b>RDECCTST</b>	23:22	rw	<p><b>Test EMEM Read Data ECC</b></p> <p>Inject a deliberate error into into the mechanism checking the ECC of data read from the Radar Memory to test the alarm generation. When set to 10b, all ECC checks should fail until this field is written with 01b. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit.</p> <p><math>00_B</math> <b>ERR0</b>, Invalid. SMCTRLSTS will be set  <math>01_B</math> <b>OFF</b>, No Error Injected  <math>10_B</math> <b>ON</b>, Error Injected  <math>11_B</math> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>
<b>RMTAERRTST</b>	25:24	rw	<p><b>Test Radar Memory Access Address Error</b></p> <p>Inject a deliberate error into into the mechanism checking for an address error condition being flagged for a access to Radar Memory to test the alarm generation. When set to 10b, all SPU accesses to Radar Memory should result in a alarm until this field is written with 01b. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit.</p> <p><math>00_B</math> <b>ERR0</b>, Invalid. SMCTRLSTS will be set  <math>01_B</math> <b>OFF</b>, No Error Injected  <math>10_B</math> <b>ON</b>, Error Injected  <math>11_B</math> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>

## Signal Processing Unit (SPU)

**Table 751 Access Mode Restrictions of SMUSER sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BPCRCTST, CINIT, RDECCTST, RIFCRCTST, RMCTRLTST, RMTAERRTST	
Otherwise (default)	r	BPCRCTST, CINIT, RDECCTST, RES, RIFCRCTST, RMCTRLTST, RMTAERRTST	Default Access Mode (Bus Error on Write)

**Table 752 Reset Values of SMUSER**

Reset Type	Reset Value	Note
Application Reset	0155 0001 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0155 0001 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

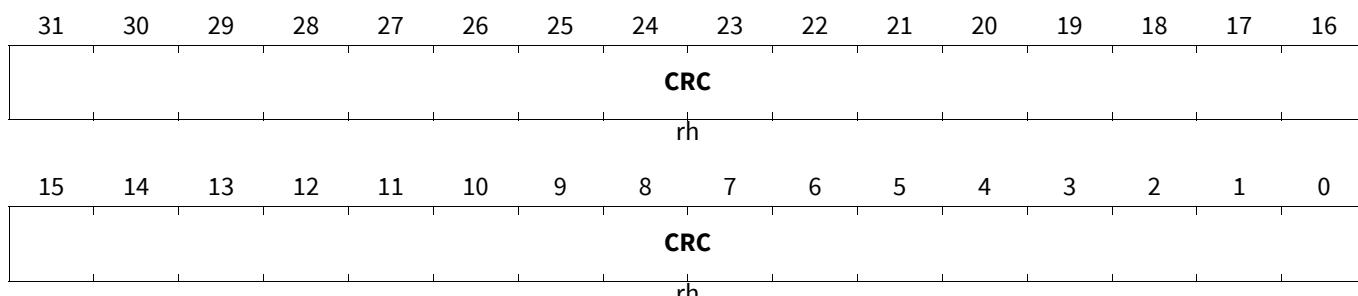
### Monitor CRC Register

CRC monitor data dependent signals within the SPU.

Allows read access to the output from one of the Monitor CRC Units

#### DATAd\_CRC (d=0-85)

**Monitor CRC Register** (00388<sub>H</sub>+d\*4) Reset Value: Table 754



Field	Bits	Type	Description
<b>CRC</b>	31:0	rh	<b>CRC</b> CRC value. Write 10 <sub>b</sub> to SMCTRL.CINIT to clear

**Table 753 Access Mode Restrictions of DATAd\_CRC (d=0-85) sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	rh	CRC	Any write access will cause bus error
Otherwise (default)	rh	CRC	Default Access Mode (Bus Error on Write)

## Signal Processing Unit (SPU)

**Table 754 Reset Values of DATA<sub>d</sub>\_CRC (d=0-85)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

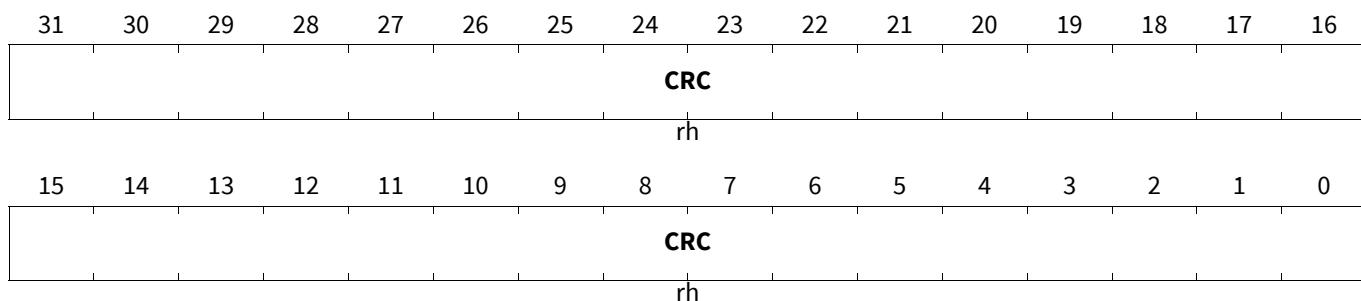
### Monitor CRC Register

CRC Monitor for data independent control signals within the SPU.

Allows read access to the output from one of the Monitor CRC Units

#### CTRLe\_CRC (e=0-24)

**Monitor CRC Register** **(00500<sub>H</sub>+e\*4)** **Reset Value: Table 756**



Field	Bits	Type	Description
CRC	31:0	rh	<b>CRC</b> CRC value. Write 10 <sub>b</sub> to SMCTRL.CINIT to clear

**Table 755 Access Mode Restrictions of CTRL<sub>e</sub>\_CRC (e=0-24) sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	rh	CRC	Any write access will cause bus error
Otherwise (default)	rh	CRC	Default Access Mode (Bus Error on Write)

**Table 756 Reset Values of CTRL<sub>e</sub>\_CRC (e=0-24)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### User OCDS Trace Control

The USROTC control register bits are only effective while the system is in debug mode. Write access requires Supervisor Mode.

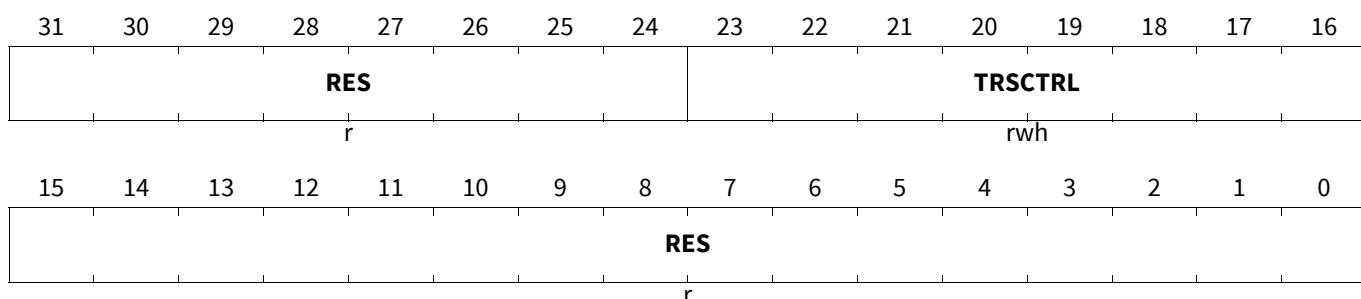
## Signal Processing Unit (SPU)

### USRRTC

#### User OCDS Trace Control

(007E0<sub>H</sub>)

Reset Value: [Table 758](#)



Field	Bits	Type	Description
RES	15:0, 31:24	r	<b>Reserved</b>
TRSCTRL	23:16	rwh	<b>Trace Control</b> This field individually controls the tracing of the eight possible output streams to the OCDS trace port. The eight bits in the field correspond to the eight Output DMA channels in ascending order. Any four of the streams can be simultaneously enabled for trace provided that the SPU and SRI are set to the same clock frequency. This field will revert to the reset value when OCDS is disabled.

**Table 757 Access Mode Restrictions of USRRTC sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rwh	TRSCTRL	
Otherwise (default)	r	RES	No Access, All accesses return Bus error
	rh	TRSCTRL	

**Table 758 Reset Values of USRRTC**

Reset Type	Reset Value	Note
Power On Reset	0000 0000 <sub>H</sub>	"Power on Reset"

### Access Enable Register 0

The Access Enable Register 0 controls access for writes to registers and configuration memory using the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <> master peripheral mapping). The SPU is prepared for a 6 bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ..., EN31 -> TAG ID 011111B.

## Signal Processing Unit (SPU)

### ACCENO

#### Access Enable Register 0

(007E4<sub>H</sub>)

Reset Value: [Table 760](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENx (x=0-31)	x	rw	<b>Access Enable for Master TAG ID x</b> This bit enables access to the SPU register addresses for transactions with the Master TAG ID x 0 <sub>B</sub> <b>RO</b> , Write access will terminate without error but will not be executed. 1 <sub>B</sub> <b>RW</b> , Write and read accesses will be executed

**Table 759 Access Mode Restrictions of ACCENO sorted by descending priority**

Mode Name	Access Mode		Description
Safety ENDINIT	rw	ENx (x=0-31)	Write access permitted during safety initialisation only. Write Accesses at other time errored
Otherwise (default)	r	ENx (x=0-31)	Default Access Mode (Bus Error on Write)

**Table 760 Reset Values of ACCENO**

Reset Type	Reset Value	Note
Application Reset	FFFF FFFF <sub>H</sub>	Application Reset

### Access Enable Register 1

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). These tags are not used in this system and so no programmable bits are provided.

### ACCEN1

#### Access Enable Register 1

(007E8<sub>H</sub>)

Reset Value: [Table 762](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															
RES															
r															

## Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>RES</b>	31:0	r	<b>Reserved</b> Read as 0, should be written with 0.

**Table 761 Access Mode Restrictions of ACCEN1 sorted by descending priority**

Mode Name	Access Mode		Description
Safety ENDINIT	r		RES Write access permitted during safety initialisation only. Write Accesses at other time errored
Otherwise (default)	r		RES Default Access Mode (Writes silently ignored)

**Table 762 Reset Values of ACCEN1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset

### OCDS Control and Status

The OCDS Control and Status (OCS) register controls the module's behaviour in suspend mode (used for debugging).

The register can only be written when the OCDS is enabled. While OCDS is disabled, the OCDS suspend control is ineffective .

#### OCS

#### OCDS Control and Status (007EC<sub>H</sub>) Reset Value: Table 764

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>RES</b>	<b>SUSST A</b>	<b>SUS_P</b>			<b>SUS</b>						<b>RES</b>				
r	rh	w		rw							r				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												<b>TG_P</b>	<b>TGB</b>	<b>TGS</b>	
												rw	rw	rw	

Field	Bits	Type	Description
<b>TGS</b>	1:0	rw	<b>Trigger Bus Select</b> Select which of the two possible trigger busses are connected to the output  00 <sub>B</sub> <b>OFF</b> , No Trigger Set Output 01 <sub>B</sub> <b>SETA</b> , Set A Output Triggers are driven onto the output 10 <sub>B</sub> <b>SETB</b> , Set B Output Triggers are driven onto the output 11 <sub>B</sub> <b>RES</b> , Reserved Do Not Use

### Signal Processing Unit (SPU)

Field	Bits	Type	Description
<b>TGB</b>	2	rw	<b>OTGB0/1 Bus Select</b> 0 <sub>B</sub> <b>OTGB0</b> , Trigger Set is Output on OTGB0 1 <sub>B</sub> <b>OTGB1</b> , OTGB1 Trigger Bus is Output on OTGB1
<b>TG_P</b>	3	rw	<b>TGS, TGB Write Protection</b> TGS and TGB are written only when TG_P is 1, otherwise unchanged. TG_P always reads 0
<b>RES</b>	23:4, 31:30	r	<b>Reserved</b>
<b>SUS</b>	27:24	rw	<b>Suspend</b> Enable Suspend of the SPU on Debug System Break. All values for this field not explicitly enumerated will cause a soft suspend of the SPU on break 0 <sub>H</sub> <b>RUN</b> , Break has no effect on SPU operation 1 <sub>H</sub> <b>HARD</b> , Break causes hard suspend of SPU
<b>SUS_P</b>	28	w	<b>Suspend Protect</b> SUS bitfield can only be updated if this bit is set to 1 for the write access
<b>SUSSTA</b>	29	rh	<b>Suspend Status</b> This field will be set to one if the SPU is suspended

**Table 763 Access Mode Restrictions of OCS sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN and Supervisor Mode and OCDS Is Enabled	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rh	SUSSTA	
	rw	TGB, TGS, TG_P	
	w	SUS_P	
Master enabled in ACCEN and OCDS Is Enabled and Supervisor Mode and write 1 to SUS_P	rw	SUS	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Supervisor Mode	r	RES, SUS, TGB, TGS, TG_P	Read Access Permitted for Supervisor Mode Accesses
	rX	SUS_P	
	rh	SUSSTA	
Otherwise (default)	r	RES, SUS, TGB, TGS, TG_P	Default Access Mode (Bus Error on Write)
	rX	SUS_P	
	rh	SUSSTA	

**Table 764 Reset Values of OCS**

Reset Type	Reset Value	Note
Power On Reset	0000 0000 <sub>H</sub>	"Power on Reset"

## Signal Processing Unit (SPU)

### OCDS Debug Access Register

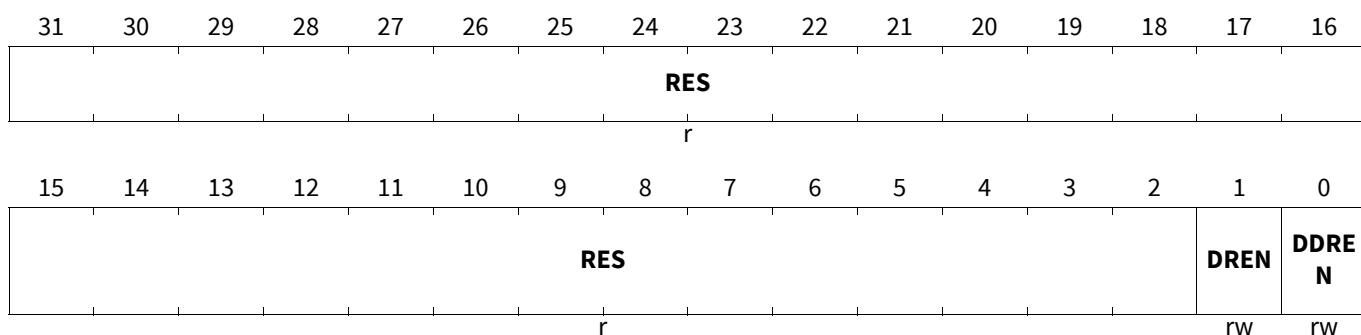
The SPU currently has no registers affected by destructive read. This register is reserved for future extension.

#### ODA

##### OCDS Debug Access Register

(007F0<sub>H</sub>)

Reset Value: [Table 766](#)



Field	Bits	Type	Description
<b>DDREN</b>	0	rw	<b>Destructive Debug Read Enable</b> If set, reads by the OCDS master to bits which would normally change state on read, do not cause the state to change
<b>DREN</b>	1	rw	<b>Destructive Read Enable</b> If set, reads to bits which would normally change state on read, do not cause the state to change
<b>RES</b>	31:2	r	<b>Reserved</b>

**Table 765 Access Mode Restrictions of ODA sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN and Supervisor Mode	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	DDREN, DREN	
Supervisor Mode	r	DDREN, DREN, RES	Read Access Permitted for Supervisor Mode Accesses
Otherwise (default)	r	DDREN, DREN, RES	Default Access Mode (Bus Error on Write)

**Table 766 Reset Values of ODA**

Reset Type	Reset Value	Note
Power On Reset	0000 0000 <sub>H</sub>	"Power on Reset"

### Kernel Reset Register 0

The KRST0, KRST1 and KRSTCLR registers are used to initiate and monitor a reset of the SPU kernel logic. The kernel reset is equivalent to a full application reset, only the registers implemented in the bus interface are not affected. All registers affected by kernel reset have an appropriate entry in the "reset values" table

Note, that if a debugger is connected, an application reset of the device will have the same effect on the KRST0 and KRST1 registers as a kernel reset. If KRST0.RST or KRST1.RST are set, they will be cleared and the KRST0.RSTSTAT bit will be set.

## Signal Processing Unit (SPU)

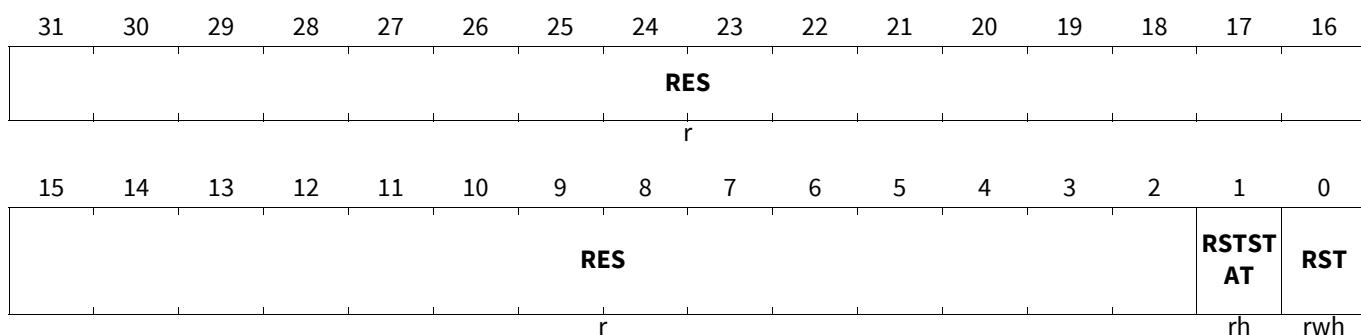
This instance of the kernel reset interface registers, KRST0, KRST1 and KRSTCLR, is an exception to the general rule that the kernel reset functionality is protected by ENDINIT. This instance is protected only by the ACCEN protection mechanism.

### KRST0

#### Kernel Reset Register 0

(007F4<sub>H</sub>)

Reset Value: [Table 768](#)



Field	Bits	Type	Description
RST	0	rwh	<p><b>Kernel Reset</b></p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to 0B) after the kernel reset is executed</p> <p><math>0_B</math> <b>Run</b>, Normal Operation  <math>1_B</math> <b>Reset</b>, Request a kernel reset of the SPU</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to 0B) after the kernel reset is executed</p>
RSTSTAT	1	rh	<p><b>Kernel Reset Status</b></p> <p>This bit indicates whether a kernel reset was executed or not. This bit is set after the execution of a kernel reset in the same clock cycle both reset bits are cleared (KRST0.RST and KRST1.RST).</p> <p><math>0_B</math> <b>Run</b>, No reset has occurred  <math>1_B</math> <b>Reset</b>, A kernel reset was executed</p>
RES	31:2	r	<p><b>Reserved</b></p> <p>Will always read as <math>0_B</math>. Should be written with <math>0_B</math></p>

**Table 767 Access Mode Restrictions of KRST0 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rh	RSTSTAT	
	rwh	RST	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	RST, RSTSTAT	

## Signal Processing Unit (SPU)

**Table 768 Reset Values of KRST0**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset

### Kernel Reset Register 1

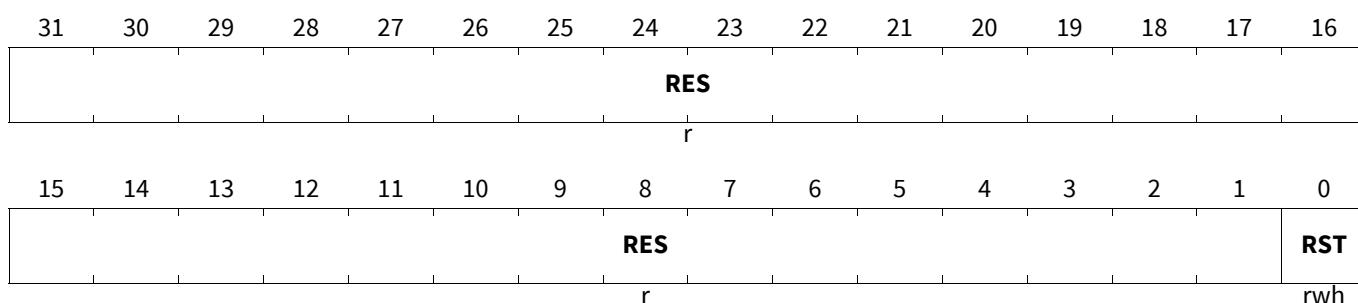
The KRST0, KRST1 and KRSTCLR registers are used to initiate and monitor a reset of the SPU kernel logic. The kernel reset is equivalent to a full application reset, only the registers implemented in the bus interface are not affected. All registers affected by kernel reset have an appropriate entry in the "reset values" table

Note, that if a debugger is connected, an application reset of the device will have the same effect on the KRST0 and KRST1 registers as a kernel reset. If KRST0.RST or KRST1.RST are set, they will be cleared and the KRST0.RSTSTAT bit will be set.

This instance of the kernel reset interface registers, KRST0, KRST1 and KRSTCLR, is an exception to the general rule that the kernel reset functionality is protected by ENDINIT. This instance is protected only by the ACCEN protection mechanism.

### KRST1

**Kernel Reset Register 1** (007F8<sub>H</sub>) Reset Value: Table 770



Field	Bits	Type	Description
<b>RST</b>	0	rwh	<p><b>Kernel Reset</b></p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to 0B) after the kernel reset is executed</p> <p><b>0<sub>B</sub></b> <b>Run</b>, Normal Operation</p> <p><b>1<sub>B</sub></b> <b>Reset</b>, Request a kernel reset of the SPU</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to 0B) after the kernel reset is executed</p>
<b>RES</b>	31:1	r	<p><b>Reserved</b></p> <p>Will always read as 0<sub>B</sub>. Should be written with 0<sub>B</sub></p>

**Table 769 Access Mode Restrictions of KRST1 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rwh	RST	

## Signal Processing Unit (SPU)

**Table 769 Access Mode Restrictions of KRST1 sorted by descending priority (cont'd)**

Mode Name	Access Mode		Description
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	RST	

**Table 770 Reset Values of KRST1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset

### Kernel Reset Clear

The KRST0, KRST1 and KRSTCLR registers are used to initiate and monitor a reset of the SPU kernel logic. The kernel reset is equivalent to a full application reset, only the registers implemented in the bus interface are not affected. All registers affected by kernel reset have an appropriate entry in the "reset values" table.

Note, that if a debugger is connected, an application reset of the device will have the same effect on the KRST0 and KRST1 registers as a kernel reset. If KRST0.RST or KRST1.RST are set, they will be cleared and the KRST0.RSTSTAT bit will be set.

This instance of the kernel reset interface registers, KRST0, KRST1 and KRSTCLR, is an exception to the general rule that the kernel reset functionality is protected by ENDINIT. This instance is protected only by the ACCEN protection mechanism.

### KRSTCLR

**Kernel Reset Clear** (007FC<sub>H</sub>) Reset Value: [Table 772](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
<b>RES</b>																
r																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
<b>RES</b>																
w																

Field	Bits	Type	Description
CLR	0	w	<b>Kernel Reset Status Clear</b> Write 1 <sub>B</sub> to clear the KRST0.RSTSTAT bit. Always reads as 0 <sub>B</sub> . 0 <sub>B</sub> <b>Run</b> , No Action 1 <sub>B</sub> <b>Clear</b> , Clear Kernel Reset Status (KRST0.RSTSTAT)
RES	31:1	r	<b>Reserved</b> Will always read as 0 <sub>B</sub> . Should be written with 0 <sub>B</sub>

**Table 771 Access Mode Restrictions of KRSTCLR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	w	CLR	

## Signal Processing Unit (SPU)

**Table 771 Access Mode Restrictions of KRSTCLR sorted by descending priority (cont'd)**

Mode Name	Access Mode		Description
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rX	CLR	

**Table 772 Reset Values of KRSTCLR**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset

## Signal Processing Unit (SPU)

### 19.5 Debug

For debugging purposes, it is possible to stop execution of a linked list. To avoid additional complexity, this can be done by software only before activating the next entry. However the debugger can stop execution using the system wide OCDS suspend functionality.

For availability reasons, the command to stop execution requires to write 3 bits,  $111_B$ , defined as STOP, to the CTRL.MODE bitfield

To reduce complexity, the Radar sequencer can only be stopped by the debugger. It is not possible to resume it. Once the OCDS suspend has been removed, the SPU should be initialised using kernel reset.

Two types of suspend are available

- Hard Suspend: this stops the kernel clock of the SPU immediately. Registers can still be read but should not be written. Config RAM cannot be accessed<sup>1)</sup>. EMEM control signals may suspend in the active state<sup>2)</sup>.
- Soft Suspend: The “GO” signals for the processing modules of the SPU are suppressed. The modules will complete, as far as possible<sup>3)</sup>, processing of the current dataset. Suspend will be acknowledged as soon as the Output Data Manager has completed all outstanding writes to Radar Memory<sup>4)</sup>.

#### 19.5.1 Trace Format

The SPU has a trace connection to the MCDS. The intention of the trace data is to allow synchronisation of SPU output events with CPU activity in the microcontroller. The MCDS has insufficient bandwidth to allow full trace of all FFT results written to EMEM so the data which can be traced simultaneously is limited. The trace logic contains sufficient buffering to allow any four<sup>5)</sup> of the eight available channels to be monitored. The following information can be output to the trace port.

**Table 773 SPU Trace Data**

Output DMA Channel	Function	Trace Data
1	FFT Results Output	Address and the first 256 bit word written for each FFT processed
2	Signal Power	Address and the first 256 bit word written for each FFT processed
3	Statistical Data	Address and data trace of all values written.
4	CFAR CA Engine Data/Local Maximum Unit Output	Address and data trace of all values written.
5	CFAR Go Engine Data	Address and data trace of all values written.

- 1) The hard suspend function operates by stopping the kernel clock of the SPU. As the config RAM arbitration logic also runs off the kernel clock, the config RAM is not accessible during hard suspend.
- 2) In this case, the EMEM tile control can be used to error the stuck access and permit debugger access to the affected EMEM tile
- 3) In some cases, data will remain in the FFT accelerator pipeline. In these circumstances the UNLOADER module will be unable to complete processing and.
- 4) If the histogram function is enabled, accesses to config RAM by the SPU may continue after soft suspend is acknowledged. This is because the FFT pipeline cannot be guaranteed to complete processing during soft suspend and is therefore not used to gate suspend acknowledgement.
- 5) For very short datasets, the bandwidth requirements for DMA channels 3 to 8 increase significantly and the number of simultaneous channels that can be traced drops. Each packet traced requires four clock cycles of the SPU clock. If the number of cycles to output the trace data exceeds the number of clock cycles required for the MATH2 unit to process the dataset, then trace data will be lost

## Signal Processing Unit (SPU)

**Table 773 SPU Trace Data (cont'd)**

Output DMA Channel	Function	Trace Data
6	Vector Add/Sum	Address and the first 256 bit word written for each FFT processed
7	sum. sum-linp, sum-antenna	Address and data trace of all values written.
8	sum of log2 power	Address and data trace of all values written.

Trace Data output is controlled by the USROTC.TRSCTRL bitfield. The eight bits in the field correspond to the eight Output DMA channels in ascending order.

### 19.5.2 Debugger events

The following event triggers are available as hardware events for the debugger via the configuration options in the OCS register.

**Table 774 OTGB Hardware Triggers (Set A)**

0	SPU Processing Starts -Trigger event defined in CTRL register has been detected
1	SPU has completed processing
2	SPU has completed a configuration load
3	Input DMA Done Signal (data block processing completed)
4	Loader Done Signal (data block processing completed)
5	Unloader Done Signal (data block processing completed)
6	ODP Loader Done Signal (data block processing completed)
7	MATH2 Done Signal (data block processing completed)
8	Output DMA Done Signal (last write completed in EMEM, set once per measurement cycle)
9	Output DMA Done (data block processing completed)
10	
11	
12	
13	
14	
15	

**Table 775 OTGB Hardware Triggers (Set B)**

0	Input DMA Run-time Error - out of range address generated
1	Loader Unit Run-time Error - out of range address generated or error converting half precision floating point input
2	Config RAM Run-time error - access aborted due to insufficient bandwidth
3	Fixed exponent format compression error - significant bits lost (UNLOADER and ODM) <sup>1)</sup>
4	Mean Power FIFO Overflow
5	Output DMA Port 1 Error

## Signal Processing Unit (SPU)

**Table 775 OTGB Hardware Triggers (Set B) (cont'd)**

6	Output DMA Port 2 Error
7	Output DMA Port 3 Error
8	Output DMA Port 4 Error
9	Output DMA Port 5 Error
10	Output DMA Port 6 Error
11	Output DMA Port 7 Error
12	Output DMA Port 8 Error
13	
14	
15	

1) This check occurs even if compression is disabled. In this case the flag will be set if data would have been lost if compression had been enabled

## 19.6 Safety Measures

The SPU has the following safety mechanisms to assist in its use in safety related applications

### 19.6.1 Hardware Safety Mechanisms

The SPU has the following autonomous hardware safety mechanisms which report failures via SMU alarms

#### 19.6.1.1 Lockstep

In products with two SPUs instantiated, there is a capability to lockstep the SPUs by writing an identical configuration set and using the same hardware trigger event to start the processing cycle. The hardware trigger could either be:

- ADC data being written from the RIF or RIF(s)
- A trigger event from the radar state machine
- A trigger event from the SPU Lockstep Module

With identical configurations and a simultaneous trigger. The SPUs will perform the same operations in the same clock cycle. This allows the address and control outputs from the SPUs to the buffer RAMS to be compared. A full description of the SPU Lockstep functionality is available in the dedicated chapter of the specification.

#### 19.6.1.1.1 Data Comparison Lockstep

To improve coverage when the same data is sent to both SPUs, a second lockstep unit also allows the data outputs to the buffer RAMs and Radar Memory to be compared. Note that this mode can only be used if both SPU instances are processing exactly the same input data and are configured to write to exactly the same Radar Memory address range.

If data is being read from Radar Memory, then the SPU1 RAM control signals will be ignored and the read data and valid signal associated with the SPU0 reads from Radar Memory will be used as the SPU1 inputs.

Data written from SPU1 will be ignored.

#### 19.6.1.2 Register CRC

The register CRC uses a 32 bit ethernet polynomial which assumes that the register contents are converted to an LSB first sequential data stream. The data stream will be generated using the registers from ID\_CONF to CTRL

## Signal Processing Unit (SPU)

inclusive. Unused register addresses, and the address of the REGCRC register, itself shall be replaced by 0000 0000<sub>H</sub> in the datastream. Additionally, the CTRL.TRIG bit and CTRL.BUSY bit should be assumed to be 0<sub>B</sub> (value when read) and the value of the PACTR.COUNT bitfield should be assumed to be 0<sub>D</sub> for calculating the reference CRC.

The polynomial shall run periodically once enabled and the CRC generated from the register contents compared with the value stored in the REGCRC.CRC bitfield. If the comparison fails, then an SMU alarm will be generated.

The Register CRC is enabled by writing 10<sub>B</sub> to the **SMCTRL**.REGCRCEN bitfield and disabled by writing 01<sub>B</sub>.

Control and comparison logic of the safety mechanism is replicated using negated logic. Any difference in behaviour between the two sets of logic will result in setting of the SMSTAT.SMCTRLSTS bit.

The alarm can be tested by writing an invalid CRC to REGCRC.CRC.

If this condition has triggered an alarm, bit 1 of the SMSTAT.SMSTAT bitfield will be set to 1<sub>B</sub>.

### 19.6.1.3 RIF Interface CRC Check

The data path between the RIF and the SPU is protected by a CRC. The CRC is checked once per ramp. The ramp size is determined from the settings of the ID\_CONF register. The check will take place when the data is written into the internal assembly buffer of the Input Data Manager. A CRC failure will result in an SMU alarm. This mechanism is controlled by the **SMCTRL**.RIFCRC bitfield.

The CRC check enabled by writing 10<sub>B</sub> to the **SMCTRL**.RIFCRC bitfield and disabled by writing 01<sub>B</sub>. Writing either of the other two possible values will result in the **SMSTAT**.SMCTRLSTS bit being set

If this condition has triggered an alarm, bit 0 of the **SMSTAT**.SMSTAT bitfield will be set to 1<sub>B</sub>.

This alarm condition can be tested by writing 10<sub>B</sub> to the **SMUSER**.RIFCRCTST bitfield.

### 19.6.1.4 Bypass Data CRC Check

The data stored by the SPU in Radar memory when bypass is enabled is protected by a CRC. The CRC is generated at the end of the measurement cycle using the same ethernet polynomial used elsewhere in the SPU and written at the next consecutive address after the last data word written. The end of the measurement cycle is determined from the settings of the ID\_CONF register. The CRC will be checked by the Input Data Manager when the read back during the bypass reload operation. A CRC failure will result in an SMU alarm.

The CRC check enabled by writing 10<sub>B</sub> to the **SMCTRL**.BPCRC bitfield and disabled by writing 01<sub>B</sub>. Writing either of the other two possible values will result in the **SMSTAT**.SMCTRLSTS bit being set.

If this condition has triggered an alarm, bit 2 of the **SMSTAT**.SMSTAT bitfield will be set to 1<sub>B</sub>.

This alarm condition can be tested by writing 10<sub>B</sub> to the **SMUSER**.BPCRCTST bitfield.

This safety mechanism can be tested by writing a short dataset with a known, bad CRC to Radar memory and triggering a bypass reload operation.

### 19.6.1.5 Radar Memory Control Signal Redundancy

All control signals controlling accesses to Radar Memory are duplicated. The duplicate signals are set to the negated value of the primary control signal. In the event of the two signals becoming inconsistent, it can be assumed that an error has occurred. An inconsistent state is detected by comparators and will cause an SMU alarm.

This check is permanently enabled. If this condition has triggered an alarm<sup>1)</sup>, bit 4 of the **SMSTAT**.SMSTAT bitfield will be set to 1<sub>B</sub>.

<sup>1)</sup> The Error flag returned by the EMEM for an access address error is also a redundant signal but, as this is already a monitor rather than a control, an inconsistency in this pair causes the **SMSTAT**.SMCTRLSTS bit to set and does not generate an alarm

## Signal Processing Unit (SPU)

### 19.6.1.6 Radar Memory Tile Access Error

Radar Memory tiles are enabled for SPU accesses by configuring the tile control mechanism in the EMEM. An SPU access can be directed to a blocked tile either by a configuration error or a fault in the SPU logic. In the event of the EMEM reporting that an access has failed because it has been mapped to a blocked tile, this mechanism can be used to trigger an SMU alarm.

The alarm is enabled by writing  $10_B$  to the **SMCTRL.RMTAERR** bitfield and disabled by writing  $01_B$ . Writing either of the other two possible values will result in the **SMSTAT.SMCTRLSTS** bit being set.

If this condition has triggered an alarm, bit 5 of the **SMSTAT.SMSTAT** bitfield will be set to  $1_B$ .

This alarm condition can be tested by writing  $10_B$  to the **SMUSER.RMTAERRTST** bitfield.

### 19.6.1.7 Radar Memory Read Data ECC

Read data returned for a Radar Memory access by the SPU is protected by an ECC computed in the EMEM from the address presented to the RAM and the data output from the RAM. This is checked by the SPU to ensure that the data is correct and has been read from the expected RAM location. A failed check will be signalled via an SMU alarm

The alarm is always enabled.

If this condition has triggered an alarm, bit 3 of the **SMSTAT.SMSTAT** bitfield will be set to  $1_B$ .

This alarm condition can be tested by writing  $10_B$  to the **SMUSER.RDECCTST** bitfield.

### 19.6.1.8 RAM ECC

RAM contents are protected by ECC. Corrupted locations will be corrected if a single bit is in error or detected if two bits are in error. Detected failures will be signalled via an SMU alarm

### 19.6.1.9 RAM Address Signature ROM

The RAM address decode is checked by a signature ROM. Failures in the address decode will be signalled via an SMU alarm

### 19.6.1.10 Access Enable

Write accesses to the registers and configuration memory<sup>1)</sup> are controlled by the Access Enable mechanism. This is configured using the **ACCEN0/ACCEN1** registers and uses the master tag used to identify the originating master of each write transaction to permit or deny access to modify the contents of the register or memory.

## 19.6.2 Software Based Safety Mechanisms

The SPU supports the following safety mechanisms which require support from software. Failures are to be reported by the software rather than via the SMU

### 19.6.2.1 Software Based Self Test

A gap occurring between application processing operations can be used to run self test patterns through the SPU to detect faults in the SPU hardware and the interfaces between the SPU and the Radar Memory.

<sup>1)</sup> The configuration memory is treated as module configuration information rather than system memory and it is therefore appropriate to apply the same protection as for the registers.

## Signal Processing Unit (SPU)

### 19.6.2.2 SPU Execution Time Check

A class of faults exist in the SPU hardware which could cause a deadlock of the SPU control hardware. A software check should be provided which does a “sanity check” on the time needed for the SPU to complete execution and flags an error if execution does not complete within the expected time. This check would also detect faults in the MMIC or MMIC<->uC interface which cause an interruption in the data stream and therefore a stall in the SPU processing.

### 19.6.2.3 Configuration Memory Content Check

The contents of the configuration memory should be checked once written to ensure that all data has been transferred to the memory correctly

## 19.6.3 Hardware Functionality Supporting Software Safety Mechanisms

The following hardware functionality exists within the SPU to increase the effectiveness of the Software Based Safety Mechanisms

### 19.6.3.1 Monitor Counters

The SPU contains counters which allow the correct operation of critical operations to be checked. These track events such as reads and writes from the Radar Memory and load and unload operations from the FFT accelerator. The monitor counters can be checked by software to ensure that the correct number of operations have been performed.

The Monitor Counters are mapped into the SPU register space as read only registers.

The Monitor Counters have an associated control register which allows the Monitor Counters to be initialised. This register can be used for fault injection. The register provides an address field (CNTCLR.SELECT) to select one of the Monitor Counters and a control field (CNTCLR.CNTTST) used to enable fault injection. When fault injection is enabled, the next read to the selected Monitor Counter register will return data with one or more bits flipped. The value of CNTCLR.CNTTST will return to OFF once the selected register has been read. Writing OFF before the selected register has been read will disable the error injection.

Writing any value other than ON or OFF is invalid and will set the SMSTAT.SMCTRLSTS bitfield.

### 19.6.3.2 Monitor CRC Units

The SPU contains CRC units used to provide a method of checking the activity on key internal interfaces. These include:

- The internal RAM interfaces of the FFT accelerator
- Buffer Ram Interfaces
- Input Data Manager EMEM Interfaces
- Output Data Manager EMEM Interface
- Configuration RAM Interfaces

The CRCs shall be separated into two blocks in the registers space.

- CRCs which are independent of the data being processed. These are enabled using **SMCTRL**.CTRLCRCEN
- CRCS which are dependent on the data being processed. These are enabled using **SMCTRL**.DATACRCEN

The first can be used for providing some detection of soft errors occurring during execution as well as during software based self test. The second can only be used while processing known data during software based self test.

The CRC units are mapped into the SPU register space as read only registers.

## Signal Processing Unit (SPU)

The CRC Units have an associated control field (**SMUSER.CINIT**). The CRC units are initialised to the reset value by writing 10<sub>B</sub> to **SMUSER.CINIT**.

**Note:** *Enabling in-line CFAR or local maximum based bin rejection will change the address sequence at the ODM output. This affects the CRC accessed via the CTRL2\_CRC register. If the control based CRC is being used then either the CTRL2\_CRC register must be excluded from the CRC checks or dynamic bin rejection based on the data being processed must not be used.*

**Note:** *Using the CRCs associated with the RAM interfaces of the FFT accelerator requires that the SPU be reset before CRC calculation starts. This is necessary because the address generators of the FFT accelerator are only initialised by reset. Otherwise they retain the state they reached at the end of the previous processing operation. A kernel reset before starting CRC calculation is recommended.*

### 19.6.3.3 Redundant Control Logic

All control and status state bits (flip-flops) implementing safety mechanism functionality are duplicated. The duplicate flip-flops are used to store the negated value of the bit. This can be observed directly in the SMCTRL and SMUSER registers. In the event of the state of these bits becoming inconsistent, it can be assumed that either a programming or soft error has occurred. An inconsistent state is detected by comparators and can be observed by polling the SMSTAT.SMCTRLSTS bit.

### 19.6.4 SMU events

The following table lists the events that can trigger an SMU alarm. See [Section 19.6.1](#) for a description of the hardware safety mechanisms.

**Table 776 SMU Events**

Event	Definition	Source	Comments
uncorrectable ECC error in Radar memory	Read data from a Radar Memory RAM has been corrupted	MTU	One of Multiple alarms will be generated based on MTU configuration
uncorrectable ECC error in a RAM internal to the SPU	Read data from an SPU RAM has been corrupted		
compare error	when using lock step, an external comparator is comparing the outputs of the 2 Radar SPU.	SPU Lockstep	

## Signal Processing Unit (SPU)

**Table 776 SMU Events (cont'd)**

Event	Definition	Source	Comments
Register CRC check	Computed Register CRC does not match pre-programmed value	SPU	Alarm Mapping is defined in the SMU Specification
Bypass Data CRC check	Computed bypass data CRC does not match the value read from memory		
Radar Memory Control Logic Failure	The control signals for accessing Radar Memory are implemented as redundant logic. This alarm condition indicates that the SPU has detected an inconsistency in the redundant logic		
RIF Data CRC Check	CRC computed from RIF data stream does not match expected value tagged onto the end of the ramp data		
Radar Memory Access Address Error	An access to the Radar Memory has been mapped to a tile that has not been enabled for SPU accesses by writing to the EMEM tile control registers		
Radar Memory Read Data ECC Check	The read data for Radar Memory accesses is returned with an ECC computed from the access address and data. This is checked and an alarm set if the check fails		

Note: *the SMU allows the user to define the action that will be taken as a result of these events (see SMU specification).*

### 19.6.5 Safety assumptions

The safety assumptions for the Radar signal processing unit are described here.

#### 19.6.5.1 Safety assumptions and safety goals

There are 2 safety cases

- Radar with 1 Radar SPU only for low end applications
- Radar with 2 x Radar SPUs for mid range and high performance radar

##### 19.6.5.1.1 Case1 = Radar with 1 Radar SPU only for low end applications

In the worse case, the Radar/ECU is involved in decision making for emergency brake or active lane assist, where at system level, customers may find it appropriate to use 2 different sensors to confirm the decision.

The other application use case considered would be where the results from the Radar/ECU application are used only to provide advisory information to the driver.

## Signal Processing Unit (SPU)

### 19.6.5.1.2 Case2 = Radar with 2 Radar SPUs only for mid to high end Radar

The radar/AURIX™ here is involved at least in decisions for directly implementing emergency braking or active lane steering without driver intervention. Again, the application may be required to use sensor redundancy at the system level.

### 19.6.5.1.3 Case3 = Radar + sensor fusion with 2 Radar SPUs only for mid to high end Radar

From the Radar/AURIX™ perspective, this case is the same as above.

In addition to this, the TriCores in the Radar/AURIX™ will be involved in doing sensor fusion in order to support high integrity decisions based on data coming from the radar/SPU and independent data coming from an external sensor. From the AURIX™ perspective, this use case is similar to the sensor fusion explained separately.

## 19.6.5.2 Radar Application assumptions

The following constraints can be used in an application in order to implement application oriented safety mechanisms.

### 19.6.5.2.1 Case 1: A decision is never taken on single ADC acquisition

This is the basic principle of FFTs for Radar: the FFT is integrating the noise measured by Radar where random variations are filtered out and where only consistent signal variations are kept.

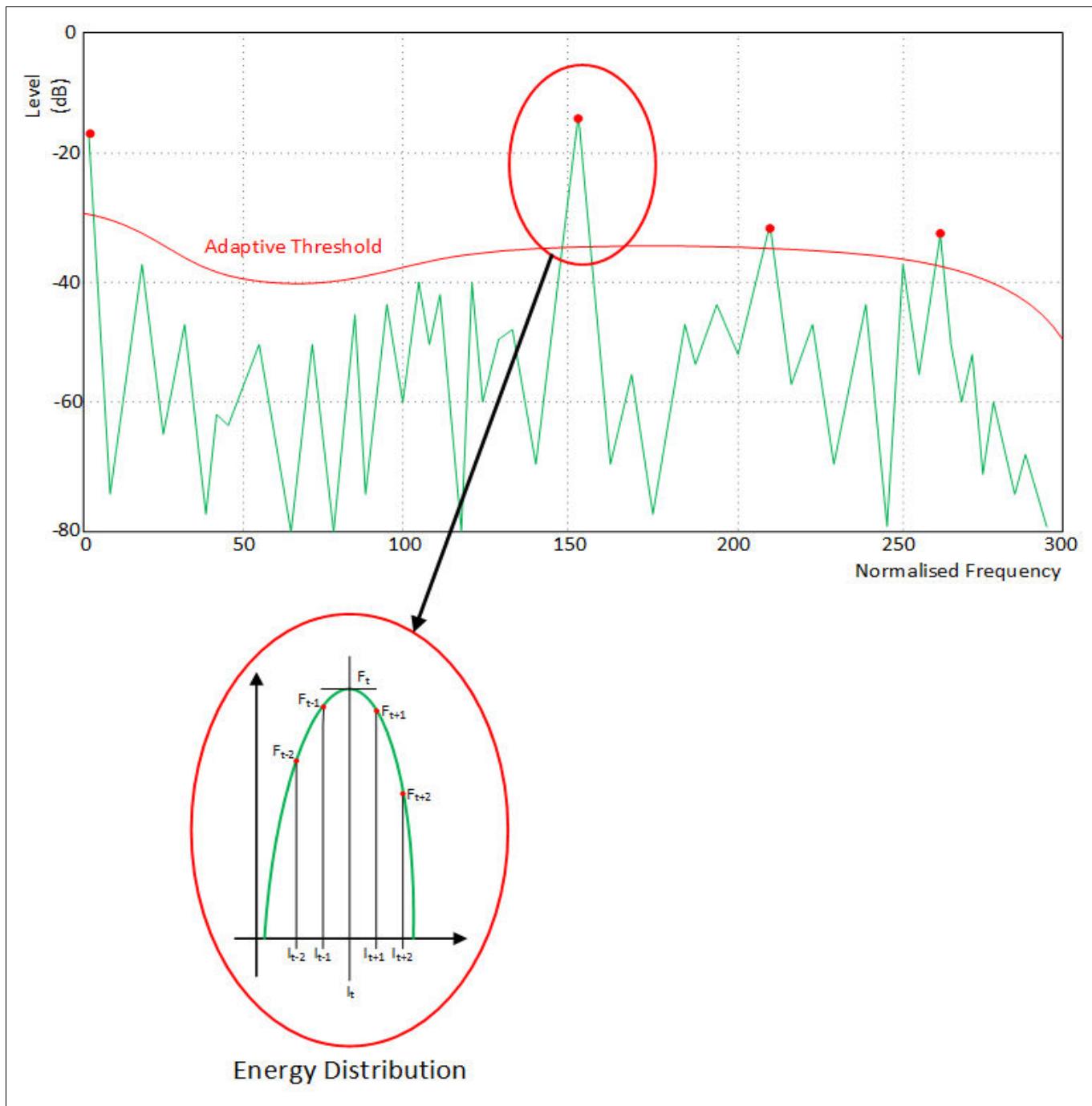
### 19.6.5.2.2 Case 2: A decision is never taken on a single FFT computation (1st stage and 2nd stage FFTs)

- Automotive Radar are now based on multiple antenna organized in phase array. So,
  - Antenna are separated by a distance
  - each antenna has same receiving lobe as the others
  - the difference between antenna is the phase of the signals: this phase difference allows the application to derive the angle of the object from the antenna.

### 19.6.5.2.3 Case 3: FFT peaks are never isolated

The figure below is showing that in fact, energy is distributed to adjacent FFT BINs so that a FFT peak is never isolated

## Signal Processing Unit (SPU)



**Figure 256 Example of FFT Result Energy Distribution**

A safety mechanism can be implemented to check if an FFT BIN is isolated to reject any isolated FFT BIN (where isolated means that the signal power on adjacent BINs is not high enough and where high enough is defined by application criteria).

Target acquisition is never taken from single acquisition

- this is definitely true for low end (see safety case1)
- this is not true for high performance radar.

Still, even if decision is taken from single acquisition, if we are allowed to assume that the RF device that is doing all preprocessing until the Analog to Digital Conversion is working fine, then, as we have a radar based on multiple

## Signal Processing Unit (SPU)

antenna with phase difference, an FFT peak leading to target decision is based on valid signal detected by multiple antenna. So

- there should be peaks detected on more than 1 antenna
- each peak on each antenna should meet the energy distribution criteria explained before

### 19.7 Use Cases

This section details assumed use case restrictions for the use of the SPU

#### 19.7.1 Use of FFT Clock Division (CTRL.DIV)

This bitfield is used to control the data throughput through the FFT engine. It is intended to be used to ensure that the FFT data is not passed to the ODP faster than it can be processed by the MATH2 unit. Although the MATH unit can exert back pressure on the UNLOADER to prevent overruns, this back pressure is then fed back to the LOADER and IDM. This is not an issue if data is being read from the Radar Memory but could potentially cause unpredictable slipping of input ramps if the data source is the RIF interface.

If data is being read from the RIF and ramp skipping is either intended or expected to be possible, the recommendation is that the CTRL.DIV bitfield should be set so that the FFT always takes longer to process the data than the MATH2 unit.

#### 19.7.2 In Place FFT

One of the key requirements for using in place FFT is that the amount of data written should not exceed the amount of data read. If this requirement is not met, then the write address generator will overtake the read address generator and start to write to locations that have not yet been read. Some particular things to watch out for

- Padding in the loader, this can increase the size of each FFT
- Bin Loop Repeat Values when using transpose mode. The output will be padded to a multiple of 32 bytes. If the bin loop repeat value and sample size result in an amount of data being read that is not a multiple of 32 bytes, then the padding will result in an increase in data size. This is not an issue when not using transpose mode as each dataset in the input data is required to be 32 byte aligned. This will result in enough free space to allow for the padding.
- Double Pass Mode is almost certain to increase if not double the data size if the FFT results are written on both passes

It is also a requirement that the Input Data Manager be configured such that each RADAR memory location is only read once in the measurement cycle. i.e. bandwidth optimisation is configured that all data samples in the 32 byte RADAR memory word are used.

#### 19.7.3 In Place FFT with ADC Data

The “In Place FFT” can be used even if the data source for the SPU is set to RIF. The Output Data Manager will use the register values in ID\_RM\_\* to compute the write addresses for the FFT results as if the SPU data was being read from EMEM. However, there are important exceptions which have to be taken into consideration

- The value of ID\_RM\_CONF.PM is not respected. The address is always generated as if this bit is set to 0<sub>B</sub>. This means that any addressing scheme which requires “Integration Mode” is not available
- If the BLR is programmed with greater value than the number of samples per line, the minimum difference between two consecutive emem write addresses will be equal to BLO/sample\_size (from ID\_RM\_CONF.FORMAT). In some cases this will mean that the addressing jumps/is not consecutive.

## Signal Processing Unit (SPU)

- For example, if BLR > 4 for COMPLEX 32 format (8 bytes) – that is 4 samples per line – and BLO is 32 bytes (256bits – 1 EMEM line), address will jump 4 lines.(BLO/No.of bytes per sample).
- It is recommended that the ID\_RM\_CONF.TRNSPS bitfield be constrained to 0<sub>B</sub>.

### 19.7.4 ODM/MATH2 Dataset Sizes

The MATH2 unit is implemented to process a minimum dataset size of eight samples. Units such as CFAR and Local Max require a larger minimum number of samples to function. Use the padding function in the LOADER module to maintain an optimum dataset size.

**Note:** *As the data block size decreases, the time needed for setup and configuration of the MATH2 unit becomes significant compared to the processing time. Performance may therefore decrease as the data block size becomes very small. For trivial dataset sizes (e.g. single samples), the SPU may not function as intended. It has been assumed that operations on datasets this small would be more efficiently performed by the CPU(s).*

### 19.7.5 In-line CFAR

There is a restriction on in-line CFAR operation for the TC39x A-step. When using double pass mode, only a single CFAR result is available for the Bin Rejection Unit. This is calculated from the first FFT of the first pass. The SPU cannot store a separate CFAR result for the first FFT of the second pass. Any configuration option related to the second pass in-line FFT result will only apply to data written to RADAR memory. This restriction does not apply to any other devices than the TC39x A-step. In the other devices, a second results register is available for the CFAR (and Local Max Unit) results which allows a different mask to be applied in both passes when double pass mode is enabled.

### 19.7.6 CFAR Configuration

Valid values should always be used to initialise the CFAR configuration registers even if it is not intended to use the CFAR engine affected by the register. This is intended to eliminate the possibility of an unexpected, invalid register setting causing an unintended output from the CFAR module.

### 19.7.7 Non-Coherent Integration

A path should always be enabled for the NCI output if the NCI function is enabled. i.e. either the CFAR or Local MAX units should be configured to use the NCI output (CFARCTRL.CFAR\_EN set to CFARN or LCLMAXN) or the direct output of NCI data to Radar memory should be enabled (NCICTRL.FORMAT=ON).

### 19.7.8 FFT Data Output Path

Do not enable in-line mode for the CFAR or Local MAX units without setting ODP\_CONF.MODE to ON

### 19.7.9 Using SUMCTRL.SUMMODE=SUMANT with unconstrained ILR value

SUMCTRL.SUMMODE=SUMANT will cause the MATH2 unit to sum across the FFTs in the output buffer memory to produce a combined FFT output (coherent integration). The use case for this is when the output buffer memory contains one FFT result per antennae. This requires an alignment between the maximum number of antennae (8) and the maximum number of FFTs which can be stored in the output buffer memory (8).

Setting up a datablock with one FFT per antennae requires that the inner loop offset of the IDM is used to step across antennae data. There are multiple IDM configurations where this isn't the case but there is no hardware lock to ensure that coherent integration is not used for these configurations. In these cases, setting

## Signal Processing Unit (SPU)

SUMMODE=SUMANT will still cause one result to be calculated per datablock even though the datablock does not contain one full iteration of the inner loop.

The behaviour of the Simulink model differs from the hardware in that the model will produce a calculation result which uses all results from the inner loop iteration, however large the inner loop repeat value. This is not possible for the RTL as it does not contain the storage needed for the intermediate results generated after the processing of each dataset.

### 19.7.10 Kernel Reset and Lockstep

In the event that it is required to perform a kernel reset of the SPU while lockstep is enabled, the SPU lockstep must be disabled before triggering the kernel reset to prevent spurious alarms being generated. This is because it is impossible to simultaneously trigger the kernel reset in both SPUs so a comparator mismatch is unavoidable.

A kernel reset must be performed before enabling lockstep if either of the SPUs has been used for data processing. This is needed to initialise the address generators in the FFT accelerator to a known state.

### 19.7.11 Supported Clocking Modes

The ADAS subsystem has the following operating restrictions

- $f_{\text{BBB}}$  must be half the SPU operating frequency  $f_{\text{ADAS}}$
- $f_{\text{SRI}}$  must be constrained relative to  $f_{\text{BBB}}$ . Ratios of 1:1 and 1:2 are supported by the EMEM.  $f_{\text{BBB}}$  cannot run faster than  $f_{\text{SRI}}$ .
- Trace will not be possible unless  $f_{\text{SRI}}$  is equal to  $f_{\text{ADAS}}$  however normal operation of the SPU will still be possible.

Assuming that the SPU is run as fast as possible (300 MHz for the TC39x), this gives two primary use cases.

- Use Case 1
  - ADAS Clock: 300 MHz
  - SRI Clock: 300 MHz
  - BBB Clock: 150 MHz
  - SPB Clock: 100 MHz
- Use Case 2
  - ADAS Clock: 300 MHz
  - SRI Clock: 150 MHz
  - BBB Clock: 150 MHz
  - SPB Clock: 75 MHz

*Note: It is unlikely that it will be possible to configure a supported clock setup until the PLL is locked. It is recommended that it is not attempted to use the SPU until a stable PLL lock is confirmed.*

### 19.7.12 RAM Initialization

The **FFT Accelerator** uses RAM instances to implement much of its internal storage requirements. Due to details in the implementation, some of the locations in these RAM instances can be read before being written with valid data. While this does not affect the results of the FFT calculation as the data is discarded, it can trigger some of the safety mechanisms in the RAM support hardware causing spurious SMU alarms.

In order to prevent this, if the alarms are enabled for the application, the RAMs should be initialised via the MTU interface before first use of the SPU.

## Signal Processing Unit (SPU)

### 19.7.13 Two SPU Instances Writing to the Same Radar Memory Tile

For TC39x derived devices, the two SPU instances must be configured to write to different tiles in the Radar Memory otherwise writes occurring in the same clock cycle will result in the SPU1 access being terminated with an error. The exception is if the device is configured for full lockstep of the SPU instances. In this case the SPU1 write is ignored and the Radar Memory is updated with the data from the SPU0 write.

For TC35x derived devices, this constraint is partially removed and the SPUs can be configured so that the FFT results from both SPU instances can be written to the same tile in the Radar Memory.

In the TC35x derived devices a single FIFO stage for SPU1 writes has been introduced. This FIFO allows SPU1 writes to be delayed by a single clock cycle so that writes occurring in the same clock cycle no longer collide. This requires that both SPU0 and SPU1 accesses to the tile use at most 50% of the available memory bandwidth and that writes from each SPU instance never occur in consecutive clock cycles. These constraints describe the writing of FFT results (PORT1 of the Output DMA Engine Data Streams as described in [Section 19.3.5, “MATH2 Unit” on Page 57](#)). When written as 32 bit precision, complex data, four FFT result bins can be fitted into each 256 bit, Radar Memory word. This means that highest throughput requirement is that one word of FFT results is written every four SPU clock cycles or every two Radar Memory clock cycles and that there is always at least one Radar Memory clock cycle before the next word of results is available to be written.

In the event of an overflow in the single stage FIFO, the affected SPU1 access will be errored.

Writing FFT results from both SPU instances to the same memory tile is the only supported use case.

### 19.7.14 Performance of the SPU

SPU performance for any given configuration is complicated by the number of configuration settings that can affect the number of clock cycles needed for each operation to complete. Generally, for simple configurations such as generating FFT results and writing them to memory, the SPU performance can approach 1 clock per bin for large datacubes. For smaller datacubes, the latency and configuration overheads for each stage in the processing pipeline become more significant. Also enabling multiple processing pipelines in the MATH2 unit or double pass processing mode can also significantly limit performance.

**Note:** *This section describes the main features of the SPU, as implemented, that can impact performance. Due to the complexity of the SPU not all configuration options that will impact performance are fully described.*

For the purposes of estimation, the SPU can be considered as three, independent pipeline stages

- The Input Data Manager (IDM)
- The LOADER/FFT/UNLOADER stage
- The MATH2 Units and Output Data Manager

The overall time to execute the configuration can also be split into three parts

- Pipeline Load Latency: The time taken to load the initial buffer RAM and trigger the FFT
- Data Processing Time: The time taken for all the data to be processed by the SPU
- Pipeline Unload Latency: the time taken to flush the MATH2 and ODM after the final FFT is processed

The total execution time for the configuration is derived by adding together all three times.

The SPU pipeline stages can sustain a processing rate in terms of clocks per bin but also incur a setup overhead for each time the buffer RAMs at the input or output of the pipeline stage are switched. In order to calculate the overall processing speed of the SPU, the time taken for each pipeline stage to process the data in the buffer RAM between each switching event needs to be calculated to determine which processing element is the slowest and therefore determines the achievable performance.

## Signal Processing Unit (SPU)

Generally, the number of times each stage of the pipeline is executed is aligned. So one IDM execution leads to one LOADER/FFT/UNLOADER execution leads to one MATH2/ODM execution. The exceptions are:

- when  $ID\_RM\_CONF.TRNSPS = 1_B$  and  $ID\_RM\_CONF.PM = 1_B$ . In this case, the  $ID\_RM\_CONF.BLOCKS$  parameter will cause the IDM to load multiple datablocks in a single execution. The second and third stage elements of the processing pipeline will therefore execute  $ID\_RM\_CONF.BLOCKS$  times for each execution of the IDM.
- When “Double Pass” is enabled ( $DPASS\_CONF.EN = 1_B$  and  $DPASS\_CONF.SWITCH = 0_B$ ). See [Section 19.7.14.5](#) for more information.

### 19.7.14.1 Input Data Manager Performance, RIF Data

The IDM operates either with the RIF as data source or the RADAR Memory (EMEM). If data is sourced from the RIF, the performance is likely to be limited by the sample rate of the internal ADCs or external RF front end, provided that the ADAS clock is set to the maximum possible frequency of 300 MHz.

The IDM itself can write up to 128 bits per clock cycle to the buffer memory. This is enough for 8 bins per clock of ADC data as a maximum data size of 16 bits is supported by the RIF interface.

Each RIF interface is 32 bits wide so can transfer 2 bins per clock. When configuring the ADAS clock used to clock the SPU and RIF, a frequency must be chosen so that the bandwidth available on the interface between each RIF and the SPU is sufficient to transfer all data received on the LVDS interface within the 2 bin per clock limit.

### 19.7.14.2 Input Data Manager Performance, EMEM Data

The IDM operates either with the RIF as data source or the RADAR Memory (EMEM). If the data is sourced from EMEM, then the IDM can read 256 bits of data every two clock cycles. As the maximum supported data size is 64 bits (32 bit precision, complex data), the IDM can maintain 2 bins per clock read performance. This is matched by the write performance which is 128 bits per clock or 2 bins per clock.

This holds as long as

- access to the EMEM is not contended
- IDM bandwidth optimisation is functioning optimally for performance

#### Notes on conditions affecting performance

1. Access contention applies when multiple SPU I/O channels are attempting to access the same tile in EMEM. For the specific case of in-place FFT, it is assumed that the SPU instance will be reading and writing from the same memory tile for at least some of the time. In this case, writing the FFT results will require 50% of the available EMEM bandwidth. The IDM can achieve 1 clock per bin performance using the other 50% available EMEM bandwidth. Consequently, the SPU can be configured for in-place FFT without affecting performance provided that tiles used for FFT results are not used for other SPU output channels.
2. The second condition will not be the case if  $ID\_RM\_CONF.TRNSPS = 1_B$  and  $ID\_RM\_CONF.PM = 1_B$ . In this case, the  $ID\_RM\_CONF.BLOCKS$  will need to be set to the correct value to ensure that all data read from EMEM is used for performance to be maintained. Even in this case, if the overall data size is small (e.g. zero or one buffer RAM switches needed to process all the data), the fact that so much of the data has to be read by the IDM to load the pipeline may mean that the pipeline load latency will be sufficiently large to degrade performance below 2 clocks per bin for executing the configuration
3. For configurations where the SPU is taking data from the RIF interface, the SPU performance can be limited by the rate at which data is fed via the RIF interface rather than the SPU configuration options. “Bypass Reload” uses the RIF interface so this consideration also applies when data is being read from EMEM when  $ID\_CONF2.BYPASS = 1_B$  and  $ID\_CONF2.BPRLD = 1_B$
4. if  $ID\_RM\_CONF.TRNSPS = 1_B$  and  $ID\_RM\_CONF.PM = 1_B$ , then performance of 2 clocks per bin may not be achieved if the IDM bandwidth optimisation is not possible. If all the data read from EMEM is not used, the excess data read and discarded still consumes EMEM bandwidth and can push the usable data fetched to less than one bin every

## Signal Processing Unit (SPU)

*two clocks.*

*e.g. The SPU can read one EMEM word every two SPU clocks. If only one usable sample can be retrieved from the fetched word the IDM performance can never be better than 1 bin per 2 clocks.*

### 19.7.14.3 LOADER/FFT/UNLOADER Performance

The performance of the LOADER/FFT/UNLOADER is primarily defined by the setting of the CTRL.DIV bitfield which sets the rate at which the LOADER pushes data into the FFT accelerator. There are also some secondary considerations which affect the overall time to process a datablock

- At the start of each datablock, the LOADER pipeline will take 9 clock cycles to initialise
- The latency through the FFT is  $2 * (\text{FFT size}) - 1$  clock cycles
- The UNLOADER latency is approximately 3 clock cycles
- The FFT is only flushed at the end of the run. This means that the first data of datablock n+1 is needed to flush the last of the data from datablock n from the accelerator. The amount of data required is related to the FFT size programmed. This can effect the latency of datablock n through the second stage of the pipeline and delay the buffer RAM switch which makes datablock n available to the MATH2 unit. This is of particular interest for the first datablock of the configuration as the start of MATH2/ODM processing will be delayed until the datablock 0 data is flushed from the FFT accelerator.

### 19.7.14.4 MATH2/ODM Performance

The performance of the MATH2 and ODM will be defined by the fetch rate from buffer memory. This is fixed at one bin every clock cycle. The following provisos apply

- The MATH2 runs a set up task every buffer RAM switch which takes approximately 12 clock cycles
- There are two classes of processing path through the MATH2 which require the data to be read from buffer RAM in two incompatible ways. Operations such as the FFT data output path through the bin rejection unit require the data to be read as consecutive points from each FFT result. The operations using the data integrated across antenna requires equivalent data points to be read consecutively from each FFT result stored in the buffer memory so the integrated value can be calculated. If both types of path are enabled, then this conflict is addressed by reading the data from the buffer memory twice. The first fetch will be used for the operations requiring linear addressing by FFT result. The second fetch will be performed for operations requiring the integrated result across antennae. The number of fetches required is evaluated and executed independently for each pass if double pass mode is enabled.
- Spectrum extension increases the effective size of each FFT by the twice the value of the CFARCTRL.SEWIN parameter.
- If in-line CFAR mode is specified, the first FFT of the configuration run will be fetched an additional time to establish the CFAR results bitmask. This is a one-off overhead occurred
- The MATH2 processing latency will be include between each of the data fetch operations (sequential and NCI/DBF) if both passes are enabled as the control logic will wait for the results of the first pass to complete before starting the second pass
  - The latency through the NCI module is approximately 2 clocks
  - The propagation delay through the CFAR module is approximately 10 clocks plus the window size (defined by CFARCFG2.GOSWINCELL, CFARCFG2.CASHWIN or CFARCFG.CAWINCELL)
  - The ODM will take approximately 8 clocks to flush the output buffers

## Signal Processing Unit (SPU)

### 19.7.14.5 Effect of “Double Pass” on Performance

If DPASS\_CONF.EN\_CNT = 0 and DPASS\_CONF.SWITCH = 0 then setting DPASS\_CONF.EN = 1 will cause the second and third stages of the processing pipeline to run twice for every one execution of the IDM stage<sup>1)</sup>.

If DPASS\_CONF.SWITCH = 1 and DPASS\_CONF.EN = 1, then the only effect is to switch parameter sets for every second datablock processed.

If DPASS\_CONF.EN\_CNT = 1 then the parameter sets will switch every DPASS\_CONF.COUNT datablocks.

Note: *Because of the pipelining, the MATH2/ODM stage of the pipeline will be executing the configuration defined by the BE0\_\* registers while the LOADER/FFT/UNLOADER is executing the configuration defined by the BE1\_\* registers and vice-versa*

## 19.8 I/O Interfaces

**Table 777 List of SPU Interface Signals**

Interface Signals	I/O	Description
sx_clocks		<b>SPU Clock Interface</b> SPU Clock Interface
sx_reset		<b>SPU Reset Interface</b> SPU Reset Interface
sx_prot		<b>SPU protection Interface</b> SPU protection Interface
sx_ocds_periph_ctrl		<b>SCU OCDS Peripheral Control Interface</b> SCU OCDS Peripheral Control Interface
sx_ocds_otgb01		<b>SPU OTGB Interface</b> SPU OTGB Interface
sx_scan		<b>SPU dft interface</b> SPU dft interface
spu_sfr		<b>BBB Slave Interface to the SPU Special Function Registers</b> SPU Module Configuration Registers
spu_cfg_ram		<b>BBB Slave Interface for Accessing the SPU Config RAM</b> BBB Slave Interface for Accessing the SPU Config RAM
sx_irq_spu		<b>SPU Interrupt Socket</b> SPU Interrupt Socket
sx_alarm_spu		<b>SPU Alarm socket</b> SPU Alarm Socket
emem_rd		<b>SPU EMEM read interface</b> SPU EMEM read interface
emem_wr		<b>SPU EMEM Write Interface</b> SPU EMEM Write Interface
emem_adc_wr		<b>SPU raw ADC data write interface to EMEM</b> Interface for writing unprocessed data directly to the EMEM.

1) or 2 \* ID\_RM\_CONF.BLOCKS if ID\_RM\_CONF.TRNSPS = 1<sub>B</sub> and ID\_RM\_CONF.PM = 1<sub>B</sub>. See Section 1.7.12.1 for an explanation.

## Signal Processing Unit (SPU)

**Table 777 List of SPU Interface Signals (cont'd)**

Interface Signals	I/O	Description
emem_math0_wr		<b>SPU EMEM Write Interface</b> SPU EMEM Write Interface
sx_ssh_com		<b>SPU Common SSH interface</b> SPU Common SSH interface
cfg_ram		<b>Config RAM SSH Interface</b> Config RAM SSH Interface
buf_ram		<b>SPU Buffer Ram SSH Interface</b> SPU Buffer Ram SSH Interface
fft0_ram		<b>FFT0 RAM MBIST Interface</b> FFT0 RAM SSH Interface
fft1_ram		<b>FFT1 RAM MBIST Interface</b> FFT1 RAM SSH Interface
fft2_ram		<b>FFT2 RAM MBIST Interface</b> FFT2 RAM SSH Interface
fft3_ram		<b>FFT3 RAM MBIST Interface</b> FFT3 RAM SSH Interface
sx_trace_sri		<b>SPU Trace Output</b> Trace Output for Data Processed by the SPU
INT	out	<b>SPU Service Request</b>
ERR		
safety_alarm	out	<b>SPU Alarm</b>
RIFD0(31:0)	in	<b>Radar input from RIFO</b> Radar input from RIFO
RIFDV0	in	<b>Radar Data Valid from RIFO</b> Radar Data Valid from RIFO
RIFD1(31:0)	in	<b>Radar Input from RIF1</b> Radar Input from RIF1
RIFDV1	in	<b>Radar data valid from RIF1</b> Radar data valid from RIF1
SDI0	in	<b>Done indication from SPU0</b>
SDI1	in	<b>Done Indication from SPU1</b>
SD	out	<b>SPU Done Output</b>
LS_T	in	<b>LockStep Trigger</b> Radar Safety Mechanism Trigger from Lockstep
test_point_en	in	<b>Enable signal for automatically inserted test points</b>
BBB_LD(267:0)	out	<b>BBB Domain Lockstep Data Outputs</b> BBB Domain Lockstep Data Outputs
BBB_LC(25:0)	out	<b>BBB Domain Lockstep Ctrl Outputs</b> BBB Domain Lockstep Ctrl Outputs
MAX_LD(592:0)	out	<b>Max Domain Lockstep Data Outputs</b> Max Domain Lockstep Data Outputs

## Signal Processing Unit (SPU)

**Table 777 List of SPU Interface Signals (cont'd)**

Interface Signals	I/O	Description
MAX_LC(70:0)	out	<b>Max Domain Lockstep Ctrl Outputs</b> Max Domain Lockstep Ctrl Outputs
EN_LBDS	in	<b>Enable Lockstep for BBB domain data signals (Active High)</b> Enable Lockstep for BBB domain data signals (Active High)
EN_LBDS_N	in	<b>Enable Lockstep for BBB domain data signals (Active Low)</b> Enable Lockstep for BBB domain data signals (Active Low)
EN_LBCS	in	<b>Enable Lockstep for BBB domain ctrl signals (Active High)</b> Enable Lockstep for BBB domain ctrl signals (Active High)
EN_LBCS_N	in	<b>Enable Lockstep for BBB domain ctrl signals (Active Low)</b> Enable Lockstep for BBB domain ctrl signals (Active Low)
EN_LMDS	in	<b>Enable Lockstep for Max domain data signals (Active High)</b> Enable Lockstep for Max domain data signals (Active High)
EN_LMDS_N	in	<b>Enable Lockstep for Max domain data signals (Active Low)</b> Enable Lockstep for Max domain data signals (Active Low)
EN_LMCS	in	<b>Enable Lockstep for Max domain ctrl signals (Active High)</b> Enable Lockstep for Max domain ctrl signals (Active High)
EN_LMCS_N	in	<b>Enable Lockstep for Max domain ctrl signals (Active Low)</b> Enable Lockstep for Max domain ctrl signals (Active Low)
app_stop	in	<b>Application Stop request from SCU</b> Application Stop request from SCU for sequencing application reset in the interests of voltage stability.

## Signal Processing Unit (SPU)

### 19.9 Revision History

**Table 778 Document Revision History**

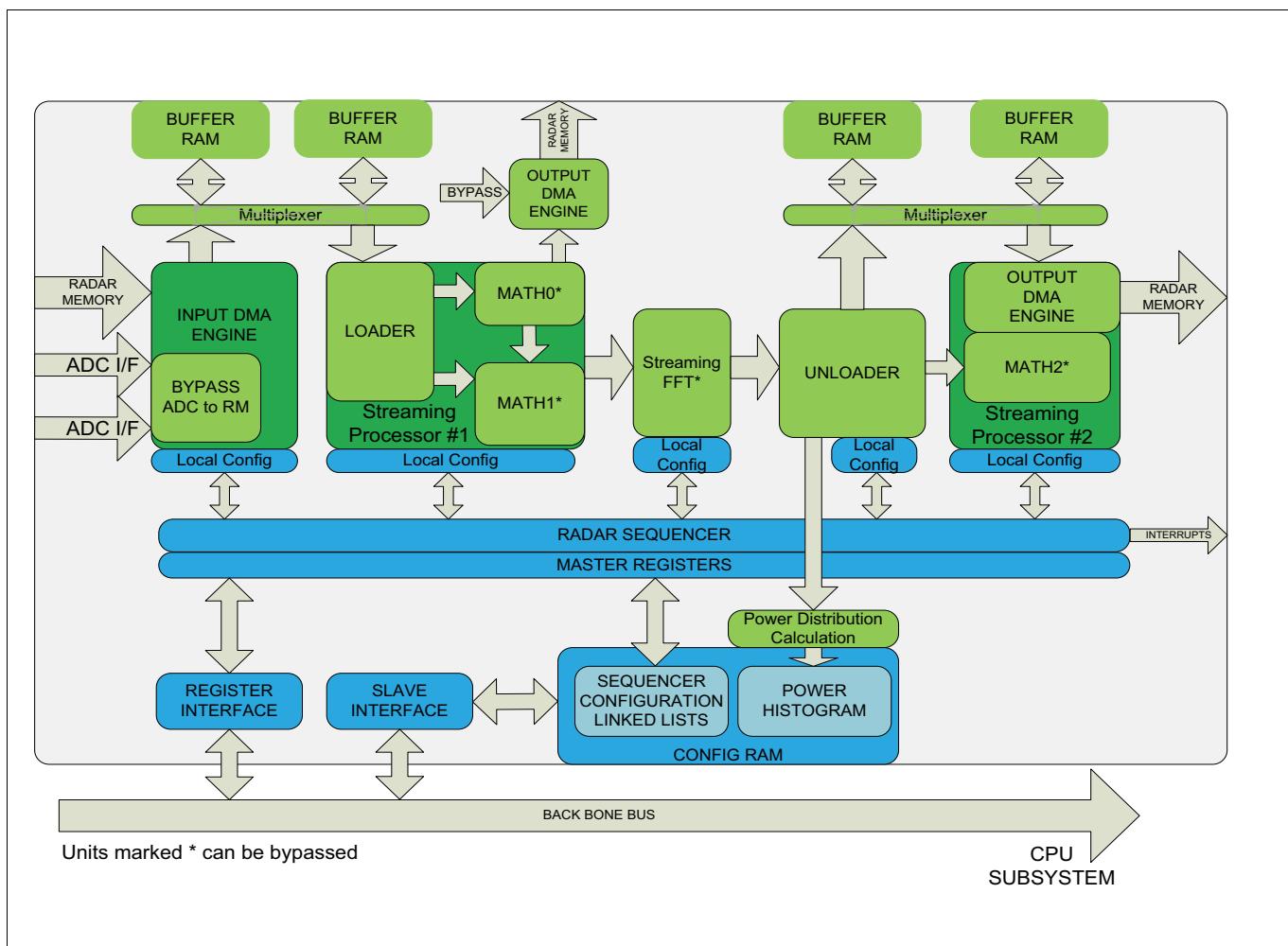
Reference	Change to Previous Version	Comment
<b>V1.1.20</b>		
<a href="#">Page 196</a>	Previous versions removed from revision history.	
<b>V1.1.21</b>		
<a href="#">Page 60</a>	<a href="#">Section 19.3.5.1.2</a> , description of algorithm for converting linear power to $\log_2$ power updated to reflect implementation.	
<a href="#">Page 138</a>	<a href="#">BINREJCTRL</a> , description of the VALUE bitfield updated	
<a href="#">Page 1</a>	<a href="#">Section 19.1</a> , description of the histogram feature updated	
<a href="#">Page 169</a>	<a href="#">ACCENO</a> , description of the EN bitfield updated. Prohibited write accesses will terminate with an error	
<a href="#">Page 47</a>	<a href="#">Section 19.3.2.1</a> , note below table corrected to put IFFT bitfield into correct register	
<a href="#">Page 187</a>	<a href="#">Section 19.7.3</a> , new use case added to cover the use of “in place FFT” for generating the write addresses for FFT results when the SPU is accepting data from the RIF(s)	
<b>V1.1.22</b>		
<a href="#">Page 10</a>	<a href="#">Section 19.2.8</a> , diagrams showing data organisation in memory after first and second stage FFT updated to correct missing bin numbers. Affected are <a href="#">Figure 231</a> , <a href="#">Figure 233</a> , <a href="#">Figure 234</a> , <a href="#">Figure 236</a> , <a href="#">Figure 238</a> , <a href="#">Figure 239</a> , <a href="#">Figure 241</a> , <a href="#">Figure 243</a> , <a href="#">Figure 244</a> , <a href="#">Figure 246</a> and <a href="#">Figure 248</a>	
<a href="#">Page 173</a> , <a href="#">Page 174</a> , <a href="#">Page 175</a>	<a href="#">KRST0</a> , <a href="#">KRST1</a> , <a href="#">KRSTCLR</a> : Description text updated to include sentence that these registers are not ENDINIT protected. This is an exception to the general kernel reset implementation.	
<a href="#">Page 88</a>	<a href="#">CLC</a> : Description for bit 2 updated from “reserved” to FDIS, “Freeze Disable”, as the bit can be written even though the “Freeze” function is not implemented and the bit has no effect.	
<b>V1.1.23</b>		
	File regenerated to align with product appendix file updates. No functional changes	
<b>V1.1.24</b>		
<a href="#">Page 191</a>	<a href="#">Section 19.7.14.1</a> , New section added to clarify dependency between LVDS data rate at RIF and required ADAS clock frequency	

## Signal Processing Unit 2 (SPU2)

### 20 Signal Processing Unit 2 (SPU2)

The Signal Processing Unit 2 (SPU2) is the second generation of the SPU, a semi-autonomous accelerator for performing Fast Fourier Transforms (FFTs) on data from one or more dedicated ADC interfaces. For the rest of this chapter this functional block will be referred to as “TC3Ax SPU”.

The TC3Ax SPU uses a three stage, streaming architecture to provide data pre-processing, FFT, and data post-processing operations. The TC3Ax SPU uses the Radar Memory to store datasets and has internal buffer memories which are used to store the data currently progressing through the processing pipeline.



**Figure 257 TC3Ax SPU Architecture**

#### 20.1 Feature List

The TC3Ax SPU has the following top level features

- Up to 2 concurrent TC3Ax SPU instances active concurrently
- A maximum achievable performance can approach 1 clock per sample processed for large data cubes. (See [Section 20.7](#) of this document for a more detailed explanation of factors affecting TC3Ax SPU throughput.)
- A 256bit bus connection to Radar memory with 2 clocks per read or write access
- Radar sequencer to manage the automatic execution of linked list where execution units are reconfigured between computation steps
- Internal configuration memory for linked lists
- 2 selectable input data flows (ADC interface or main RAM)

## Signal Processing Unit 2 (SPU2)

- The ADC Interface supports real or complex data
- Input / output precision selectable: 16bits, 32bits, half precision floating point or compressed
- Input DMA with HW transpose unit and data padding
- Multiple execution units
  - Time domain interference detection and removal
  - Windowing with complex windowing and HW support for phase demodulation
  - FFT from 8 to 2048 points
  - 2 x 1D-CFAR, thresholding units implementing 4 CA and 3 GOS algorithms
  - 1 x BIN rejection unit
  - Non coherent integration over 2 TC3Ax SPU instances
  - Digital beam forming over 2 TC3Ax SPU instances
  - Antenna signal integration for elevation
  - Basic thresholding (local maximum detection)
- RAM size optimisation by using configurable FFT strategy (in place, out of place, partially shifted)
- Output DMA with output FIFOs so that each data flow has its own FIFO and configurable DMA parameters
- A min/max/averaging unit.
- A histogram unit up to 4096 elements
- Support for up to 8 data lanes @ 600Mbit/s
- Low speed 60 Mb/s single ended IF

### 20.2 Overview

See Overview of the SPU chapter for an overview.

### 20.3 Functional Description

This section describes programming and using the TC3Ax SPU. It deals with each of the major functional blocks of the TC3Ax SPU in the order data flows through the processing pipeline.

#### 20.3.1 Input DMA Engine

The Input DMA Engine can either passively accept data pushed by the ADC interfaces (RIF or Radar Interface) or actively load part or all of an existing data cube from the Radar Memory. Its function is to structure the data in the Buffer RAM into FFT data sets. To do this, it uses the following general control data as well as control information specific to the data source used.

**Table 779 Input DMA Unit General Parameters for all Input Sources**

Parameter	Definition	Comments
Data Source	ADC (RIF), Bypassed data or Radar Memory data	<b>ID_CONF2.SRC</b>
Source RIF	Select which ADC interface (RIF) data are processed	<b>ID_CONF2.SRCRIFm</b> Note: If SRC==RIF and SRCRIFm are all zero the behaviour is undefined
start mode	trigger mode for the TC3Ax SPU ( <b>CTRL.MODE</b> ). Options are defined in <a href="#">Table 822, CTRL.MODE Value Definitions</a>	

## Signal Processing Unit 2 (SPU2)

### 20.3.1.1 Load ADC data from the RIF

When loading data from the RIF or RIFs, processing mode is always default and the input data format and number of antennae must be aligned with the RIF configuration settings. In addition the following operating modes can be configured

**Table 780 Input DMA Unit Specific Parameters for ADC Data**

Parameter	Definition	Comments
Ramps per Measurement Cycle	Number of ramps to be acquired after trigger.	Excess ramps will be ignored unless the TC3Ax SPU is retriggered <b>ID_CONF.RAMPS</b>
Number of active antenna	Number of physical antennae connected to the ADC interfaces of the microcontroller	<b>ID_CONF2.ANT</b>
Input format	Signed or unsigned	<b>ID_CONF2.SIGNED</b>
Samples per Ramp or Bins per FFT	The number of sample points to be used in the input datasets for the FFT <sup>1)</sup>	<b>ID_CONF.SMPLCNT</b>
Input format	Real or complex	<b>ID_CONF2 FORMAT</b>

1) Although one is the minimum number of samples the MATH0 unit cannot function correctly with less than eight.

### 20.3.1.1.1 Principles of Operation (ADC IF)

There are some restrictions imposed on the physical configuration of the overall system by the Input DMA. The most significant of these are that, if both ADC IF<sup>1)</sup> instances are being processed by a single TC3Ax SPU:

- the configuration of the ADC IFs must be identical so that the quantity of data being fed to the TC3Ax SPU by each ADC IF is the same.
- The data sources providing data to the ADC IFs must be synchronised. This is for two reasons
  - To enable meaningful integration (or other calculations) across the antennae, the sampling time for each antenna must be aligned and kept consistent to within very small limits
  - The ADC IFs contain buffers to ensure that the data fed to the TC3Ax SPU is aligned in the same clock cycle for equivalent antenna. The alignment of data is triggered by setting a configuration option in the RIF module registers. The data provided to the ADC IFs must be sufficiently aligned to prevent these buffers overflowing. The buffering is sufficient to allow for one complete sample misalignment at the serial inputs to the ADC IFs.

When configured for internal or software trigger mode, the TC3Ax SPU will accept all data from the ADC IFs. The ADC IF in this case must maintain synchronisation to the ramps and only pass valid data to the TC3Ax SPU (although there is a limited facility to remove undesired data by truncating the dataset in the MATH1 unit).

#### Antennae (ID\_CONF2.ANT)

The number of active antennae is the number of physical antennae attached to the configured data source (ADC IF0, ADC IF1 or both). This can be up to four antennae per ADC IF (RIF).

1) The ADC IF function is implemented in the RADAR Interface (RIF) module

## Signal Processing Unit 2 (SPU2)

### 20.3.1.1.2 Bypass Function

The Bypass function of the Input DMA allows ADC (RIF) data to be both written to Radar Memory and processed in the TC3Ax SPU at the same time. The Bypass function operates independently from the Input DMA Load from Radar Memory function allowing RIF data to be bypassed at the same time as the TC3Ax SPU processes data read from Radar Memory. A single TC3Ax SPU configuration can only have ID\_CONF2.SRC set to either RIF or RM but the Bypass function requires the source to be set to RIF. To get the TC3Ax SPU to process data from Radar Memory and bypass RIF data at the same time requires two different configurations. The RELOAD MODE in the CTRL register can be used to trigger the Bypass function and the next configuration can be used to trigger the TC3Ax SPU to process data from the Radar Memory.

The RIF interfaces for which data shall be processed are identified by setting the corresponding ID\_CONF2.SRCRIFm bits. The ID\_CONF2.SRC bitfield must also be set to RIF. If CTRL.BPTRIG bit is set to a 1<sub>B</sub> and at least one of the ID\_CONF2.SRCRIFm bits is set to a 1<sub>B</sub> when the SPU is triggered the bypass function shall be triggered. The Bypass function may be triggered alone by using the RELOAD MODE in the CTRL register. The base address for writing bypassed data is defined by BYPASS\_CTRL.BPADDR. Data is written to memory in the order as it is received from a RIF. When more than one RIF is bypassed then data is written in the order it is received from the RIF interfaces and data from RIF0 shall be added to the Radar Memory word first. In this case the resulting memory word will have 32 bits from RIF0 then 32 bits from RIF1 followed by 32 bits from RIF0 and so on.

Once triggered, the Bypass function will operate until it has written the amount of data as defined by ID\_CONF.SMPCNT, ID\_CONF.RAMPS, ID\_CONF2.ANT from all RIFs selected by BYPASS\_CTRL.SRCRIFm. The Bypass function takes a copy of the values of these registers when it is triggered. If STAT.INTMSK(2) is set, then the Bypass function will trigger an ATTN interrupt when the last data has been written to Radar Memory. The Bypass function will always run to completion, independent of the state of the TC3Ax SPU, and cannot be stopped except by resetting the TC3Ax SPU. Triggering Bypass when Bypass is busy, as indicated by the MONITOR.BP\_BUSY bit, has no effect.

Data that has been written to Radar Memory using the Bypass function may be read back for processing. To do this ID\_CONF2.SRC must be set to BPRLD and ID\_CONF2.SRCRIFm must be set the same values as BYPASS\_CTRL.SRCRIFm were when the data was written. Attempting to reload data by setting ID\_CONF2.SRC==BPRLD after the Bypass function has been triggered with the same setting of BYPASS\_CTRL.BPADDR is not recommended as such a configuration would cause a race condition between reading and writing data at the same location in Radar Memory.

**Table 781 Bypass Configuration**

Parameter	Definition	Comments
Bypass trigger	When CTRL.BPTRIG is set any trigger of the SPU will trigger Bypass. In addition, Bypass can be triggered alone by CTRL.MODE==RELOAD	<b>CTRL.BPTRIG CTRL.MODE</b>
Bypass address	Base address for writing bypass data	<b>BYPASS_CTRL.BPADDR</b>
Bypass source enable	Select which ADC (RIF) data source are bypassed. At least one of SRCRIFm must be set for Bypass to be triggered	<b>BYPASS_CTRL.SRCRIFm</b>

## Signal Processing Unit 2 (SPU2)

**Table 781 Bypass Configuration (cont'd)**

Parameter	Definition	Comments
Bypass read-back	Enable reading of bypassed data	<b>ID_CONF2.SRC</b> Set to BPRLD
Bypass read-back source	Should the set the same as BYPASS_CTRL.SRCRIFm when data was written. At least one of SRCRIFm must be set. A configuration with SRC==BPRLD and these bits all zero will result in undefined behaviour.	<b>ID_CONF2.SRCRIFm</b>

### Split Processing

Bypass may be used in systems with 2 RIF instances when there is too much data from the ADCs to be handled by the SPUs in a single pass. ID\_CONF2.SRC should be set to RIF and the antennae must be distributed symmetrically between the two RIF instances.

The ID\_CONF2.SRCRIFm should be set to select one RIF (RIF0, say) so that data from RIF0 is processed. BYPASS\_CTRL.SRCRIFm should be set to select the other RIF (RIF1, say) so that the data from RIF1 is written directly to Radar Memory for later processing. The base address is stored in the BYPASS\_CTRL.BPADDR field. The first pass is configured by setting BYPASS\_CTRL.BYPASS. This will process the RIF0 data and bypass the RIF1 data. The second pass is configured by setting ID\_CONF2.SRC to BPRLD and setting ID\_CONF2.SRCRIFm to the same setting of BYPASS\_CTRL.SRXCRIFm (RIF1, say) as was used in the first pass. This will read the data from the address in the BPADDR field rather than the RIF interface so BPADDR should be left at the same value as in the first pass.

All other register settings, apart from the ODP\_BASE.BASE should be the same. In Bypass Mode, the reloaded data is treated as an ADC IF (RIF) data stream rather than a datacube in Radar Memory. So the same parameters apply as when loading ADC data directly from a RIF. The parameters used when loading from Radar Memory are not applicable.

#### 20.3.1.2 Load from Radar Memory

The following options are specific to loading data from Radar Memory:

**Table 782 Input DMA Unit Parameters for Reading Data from Radar Memory**

Parameter	Definition	Comments
Input format	16bit or 32bit precision complex data, 16bit or 32bit precision real data, IEEE 754 half precision floating point real data, IEEE 754 half precision floating point complex data, compressed complex data, 32 bit power, IEEE 754 half precision floating point power data,	<b>ID_RM_CONF2.FORMAT</b>
DMA base address	DMA base address when starting a new FFT sequence from Radar memory	Word (256 bit) address relative to start of Radar Memory <b>ID_RM_CONF.BASE</b>

## Signal Processing Unit 2 (SPU2)

**Table 782 Input DMA Unit Parameters for Reading Data from Radar Memory (cont'd)**

Parameter	Definition	Comments
DMA outer loop address offset	Offset address to be added to base address (Radar Memory address calculation)	Byte offset <b>ID_RM_OLO.OLO</b>
DMA outer loop repeat value (parameter “m”)	Number of times of outer loop execution	e.g. for 2nd stage FFT, this would be the number of antennae to process in default mode. <b>ID_RM_IOLR.OLR</b>
DMA inner loop address offset	Offset to be added to base address (Radar Memory address calculation)	Byte offset <b>ID_RM_ILO.ILO</b>
DMA inner loop repeat value (parameter “n”)	Number of times of inner loop execution	Inner loop count is incremented by one every time a complete FFT is read from memory <b>IDM_RM_IOLR.ILR</b>
Bin offset	Radar Memory address offset between adjacent bins in an input dataset	Byte offset. When reading a linear FFT from memory, this would be set to the size of one data element <b>IDM_RM_BLO.BLO</b>
Bin Offset Loop Repeat (parameter “p”)	Number of reads needed to traverse one input FFT. (number of bins in an input dataset)	The number of samples in each FFT input (dataset) being constructed <b>IDM_RM_BLR.BLR</b>
Addressing Mode	Linear or Transpose Addressing	<b>ID_RM_CONF2.TRNSPS</b>
Processing Mode	Default Mode or Integration Mode for bandwidth optimisation.	Integration Mode should be used when constructing input datasets for use with the coherent or non-coherent integration functions in the MATH2 units <b>IDM_RM_CONF2.PM</b>
Number of simultaneous data blocks from RAM	For integration mode, defines the number of concurrent information sets (data blocks) that are fetched from memory. Depends on memory organization. Number from 1 to 8.(1,2 4 and 8 supported).	Number of data blocks that will fit into the buffer memory. <b>IDM_RM_CONF2.BLOCKS</b> Add one to this field to get the number of data blocks. Supported values are therefore: 0, 1, 3, and 7.

## Signal Processing Unit 2 (SPU2)

**Table 782 Input DMA Unit Parameters for Reading Data from Radar Memory (cont'd)**

Parameter	Definition	Comments
Mapping of FFT datasets to antenna	Defines how each dataset in the buffer memory is linked to an antenna ID for windowing and other antenna specific operations	<b>ID_RM_CONF2.AM</b>
Incremental Processing Mode?	Allows one TC3Ax SPU instance to trigger another on a ramp-by-ramp basis providing a ramp fits into the buffer RAM. The IDM waits for the Incremental Trigger input from the other TC3Ax SPU before first loading the buffer RAM and after each buffer RAM switch.	<b>ID_RM_CONF2.IMODE</b>

All address fields in registers are specified in words (256 bits or 32 bytes).

At any time the byte address that the Input DMA is reading from is defined as:

```
(base address<<5) + ((outer loop offset)*m) + ((inner loop offset)*n) + ((bin loop offset)*p)
```

where **n** and **m** are as defined above and **p** is the bin being read. This will be used to generate a word address for Radar Memory by truncating bits [4:0] of the generated address. Bits [4:0] will be used to select within the 256 bit word retrieved from radar Memory.

The input DMA engine will read (at least) one complete set of FFT input data per inner loop using the Bin Offset field to increment the byte address between sample reads.

A more detailed description of the operating modes is given in the following sub-sections.

### 20.3.1.2.1 Principles of Operation (Radar Memory)

The Input DMA is designed to allow autonomous processing of a three dimensional data array stored in the Radar memory. Typically the three axes of the array would be representing FFT sample number, ramp or acquisition count and antenna. The intention is to allow assembling input sets for the FFT accelerator by stepping across the axes in any arbitrary order.

This is accomplished by constructing the read address for the Input DMA from a base address and three independent address offsets (“bin loop address”, “inner loop address” and “outer loop address”).

The term “bin loop” is used as the bin loop repeat value maps directly to the number of bins in each FFT dataset constructed in the buffer RAM. The bin loop repeat value defines the size of the input dataset.

**Note:** *The maximum number of bins supported in an FFT dataset is 2048. Setting a Bin Loop Repeat value for an FFT size greater than this (i.e. ID\_RM\_BLR.BLR > 2047) is not supported and should be avoided.*

Some assumptions are made about the organisation of the data. Specifically that:

- Each dataset in memory starts at a 32 byte aligned address. This allows the bandwidth optimisation hardware to function and is enforced by the Output Data Manager. This restriction has to be observed when using software to build data arrays in memory for reading by the Input Data Manager
- The second assumption follows from the first. If all datasets are 32 byte aligned, it follows that 2 co-ordinate axes of the data array must be multiples of 32 byte. These axes will be used to step along datasets. Only a single co-ordinate axis, the one used to step along data samples stored in adjacent memory locations, can be less than a multiple of 32 bytes and this can be set to the data sample size. However, bandwidth optimisation

## Signal Processing Unit 2 (SPU2)

will enforce the reading of data in 32 byte words, so this axis will increment by 32 bytes and the IDM will behave in an identical manner whether the increment is set to sample size or 32 bytes. Note that this axis can also be set in the IDM to be a multiple of 32 bytes but, in this case, the data read for this axis will be non-contiguous and will “jump” after every 4, 8 or 16 samples depending on the sample size.

**Note:** *The TC3Ax SPU is specified to cope with Radar Memory sizes up to 16 MiB. If programming causes it to generate an address outside the range of the available Radar Memory in the product, an error will be generated by the Radar Memory*

Each counter has an associated offset and a repeat count. Operation (neglecting the effect of bandwidth optimisation) is then as follows:

**Table 783 Input DMA Execution Flow**

1	initialise the “bin loop address”, the “inner loop address” and the “outer loop address”.
2	loop
3	The read address is computed by adding the “base address”, the “bin loop address”, the “inner loop address” and the “outer loop address”.
4	An input value is then read from the address
5	“bin loop count” is incremented
6	If { the bin loop count has reached the “bin offset loop repeat” value } then
7	“bin loop count” is reset
8	“bin loop address” is reset
9	“inner loop count” is incremented
10	If { the “inner loop count” has reached the “inner offset loop repeat” value } then
11	“inner loop count” is reset
12	“inner loop address” is reset
13	“outer loop count” is incremented
14	If { the “outer loop count” has reached the “outer offset loop repeat” value } then
15	execution has completed, exit loop
16	else
17	“outer loop address” is set to “outer loop address” plus “outer loop offset”
18	end if
19	else
20	“inner loop address” is set to “inner loop address” plus “inner loop offset”
21	end if
22	else
23	“bin loop address” is set to “bin loop address” plus “bin loop offset”
24	end if
25	end loop
26	stop

The flow above is the basic operation flow without considering bandwidth optimisation. As the Radar Memory is organised as 256 bit words and the data element occupy either 32 or 64 bits depending on precision and format, then bandwidth usage can be optimised by using as much as possible of the 256 bit word<sup>1)</sup>. The Input DMA will

## Signal Processing Unit 2 (SPU2)

attempt to do this based on register settings by “collapsing” one or more of the loops i.e. effectively performing multiple iterations of one of the loops in a single pass as described in the following sections.

**Table 784 Input DMA Loop Mapping**

Mode	Co-ordinate Mapping <sup>1)</sup>			Notes	ID_RM_CONF2 Register Settings <sup>2)</sup>
	Bin Loop Address Iteration	Inner Loop Address Iteration	Outer Loop Address Iteration		
1	<b>Sample</b>	FFT	Antenna	Linear (Default) Addressing Mode. Each Radar Memory Read fetches multiple samples for the same FFT. Integration mode can be used to build a datablock for NCI or DBF <sup>3)</sup> . Blocks must be 0 <sub>B</sub> See <a href="#">Chapter 20.3.1.2.2</a>	<b>TRNSPS=0</b> PM=0,1 <b>BLOCKS=0</b>
2	<b>Sample</b>	Antenna	FFT	Linear (Default) Addressing Mode. Each Radar Memory Read fetches multiple samples for the same FFT. Integration mode can be used to build a datablock for NCI or DBF. Blocks must be 0 <sub>B</sub> See <a href="#">Chapter 20.3.1.2.2</a>	<b>TRNSPS=0</b> PM=0,1 <b>BLOCKS=0</b>
3	FFT	<b>Sample</b>	Antenna	Transpose Addressing Mode. Each Radar Memory Read fetches samples for multiple FFT datasets. Integration mode is not supported. Blocks must be 0 <sub>D</sub> See <a href="#">Chapter 20.3.1.2.3</a> . ID_RM_CONF2.FORMAT=REAL16BIT not supported. ID_RM_CONF2.FORMAT=REAL16FP not supported.	<b>TRNSPS=1</b> PM=0 <b>BLOCKS=0</b>
4	FFT	Antenna	<b>Sample</b>	Transpose Addressing Mode. Each Radar Memory Read fetches samples for multiple data blocks. Integration mode must be set. Blocks can be set to a value other than 0 <sub>D</sub> for bandwidth optimisation See <a href="#">Chapter 20.3.1.2.4</a> ID_RM_CONF2.FORMAT=REAL16BIT not supported. ID_RM_CONF2.FORMAT=REAL16FP not supported.	<b>TRNSPS=1</b> PM=1 <b>BLOCKS=0,1,3,7</b>

1) The exceptions to this are the 16 bit precision, real only formats, REAL16BIT and REAL16FP. These formats can only be used if TRNSPS=0. If the TC3Ax SPU is configured with (FORMAT=REAL16BIT or FORMAT=REAL16FP) and TRNSPS=1, the resulting datasets will be corrupted.

## Signal Processing Unit 2 (SPU2)

**Table 784 Input DMA Loop Mapping (cont'd)**

Mode	Co-ordinate Mapping <sup>1)</sup>			Notes	ID_RM_CONF2 Register Settings <sup>2)</sup>
	Bin Loop Address Iteration	Inner Loop Address Iteration	Outer Loop Address Iteration		
5	Antenna	<b>Sample</b>	FFT	Transpose Addressing Mode. Each Radar Memory Read fetches samples for multiple FFT datasets. Integration mode is not supported. Blocks must be $0_D$ . See <a href="#">Chapter 20.3.1.2.3</a> ID_RM_CONF2.FORMAT=REAL16BIT not supported. ID_RM_CONF2.FORMAT=REAL16FP not supported.	<b>TRNSPS=1</b> PM=0 <b>BLOCKS=0</b>
6	Antenna	FFT	<b>Sample</b>	Transpose Addressing Mode. Each Radar Memory Read fetches samples for multiple data blocks. Integration mode must be set. Blocks can be set to a value other than $0_D$ for bandwidth optimisation. See <a href="#">Chapter 20.3.1.2.4</a> ID_RM_CONF2.FORMAT=REAL16BIT not supported. ID_RM_CONF2.FORMAT=REAL16FP not supported.	<b>TRNSPS=1</b> PM=1 BLOCKS=0,1,3,7

- 1) Incrementing the Offset counter increments address to next sample bin, next FFT dataset or next antenna. When iterating across samples, data is still read as 256 bit words so the address will increment by at least one, 256 bit word as long as bandwidth optimisation is operating.
- 2) Mandatory settings are shown in bold. Options where multiple values are possible are shown as a commas separated list
- 3) Provided that ID\_RM\_IOLR.ILR is less than or equal to  $7_D$ .

### Incremental Trigger

When ID\_RM\_CONF2.IMODE is set, the IDM will wait for an Incremental Trigger from another TC3Ax SPU before first loading the Buffer RAM with data and after each Buffer RAM switch. The other TC3Ax SPU must be configured with CTRL.ITRIG set to generate the Incremental Trigger when it has finished processing the contents of its Buffer RAM and has written all data to the Radar Memory. The TC3Ax SPU will keep track of the number of outstanding incremental trigger events if the other SPU is processing data faster than the SPU. Configuring two TC3Ax SPUs both with ID\_RM\_CONF2.IMODE set is not supported as they would both wait for a trigger from the other causing a deadlock.

### 20.3.1.2.2 Case 1: “Bin Offset” is used to Iterate Across Adjacent Samples

In this case, the bins in Radar Memory to be used as consecutive inputs bins for each FFT dataset are stored in consecutive addresses in Radar Memory so each 256 bit read will load four, eight or sixteen bins depending on the precision of the data. The input DMA will therefore execute four, eight or sixteen iterations of the “bin offset” loop simultaneously. As this does not require address transposition, the ID\_RM\_CONF2.TRNSPS bit must be set to  $0_B$  (LIN).

Bandwidth optimisation is not needed in this mode as all data fetched in a 256 bit read is used in the same FFT dataset.

### Buffer Memory Switching

If the ID\_RM\_CONF2.PM bit is set to IM, this signals that the data block should contain one dataset for each antenna (i.e. the Inner Loop Offset is set to step across antennae) and the Buffer Memory will be switched when

## Signal Processing Unit 2 (SPU2)

the Inner Loop Repeat Count is reached. Otherwise, the Input Buffer Memory will be switched every four (for 64 bit data) or eight (for 32 bit data) iterations of the Inner Loop or when all data has been read whichever occurs first.

**Note:** *Setting ID\_RM\_CONF2.BLOCKS to a non-zero value with ID\_RM\_CONF2.TRNSPS set to 0<sub>B</sub> is not supported and will result in undefined operation of the input DMA.*

### 20.3.1.2.3 Case 2: “Inner Loop Offset” is set to Iterate Across Adjacent Samples

This is the first of the co-ordinate transposition cases, so the ID\_RM\_CONF2.TRNSPS bit must also be set to explicitly signal the intent to use co-ordinate transposition to the logic of the TC3Ax SPU. The Bin Loop offset can be set to step either across antennae or across FFT result datasets.

The ID\_RM\_CONF2.TRNSPS bit signals that the data from each 256 bit read will be used in different FFT input datasets and should therefore be written to different areas of the buffer memory (“scattered”).

As stated, the bins in Radar Memory to be used as consecutive input bins in each FFT dataset are stored in non-consecutive addresses in Radar Memory so each 256 bit read will load four or eight bins depending on the precision of the data<sup>1)</sup> but these samples will be from different FFT input sets. The input DMA will therefore execute four or eight iterations of the “inner loop” simultaneously. This bandwidth optimisation is described in [Section 20.3.1.2.5, “Bandwidth Optimisation for Default Processing Mode” on Page 12](#).

ID\_RM\_CONF2.PM must be set to DM for this operating case as the data from each 256 bit read is all needed for a single datablock.

#### Buffer Memory Switching

The Input Buffer Memory will be switched either when the Inner Loop Count is updated. i.e. every four (for 64 bit data) or eight (for 32 bit data) iterations of the Inner Loop or when all data has been read.

### 20.3.1.2.4 Case 3: “Outer Loop Offset” is set to Iterate Across Adjacent Samples

This is the second of the co-ordinate transposition cases, so the ID\_RM\_CONF2.TRNSPS bit must also be set to explicitly signal the intent to use co-ordinate transposition to the logic to the logic of the TC3Ax SPU. Either the Bin Loop Offset or Inner Loop Offset will be set to step across the FFT datasets stored in the Radar Memory. However in both of these cases, consecutive samples to be used as inputs to the FFT are stored in non-consecutive addresses in Radar Memory so, while each 256 bit read will load four or eight bins<sup>1)</sup> depending on the precision of the data, these samples will be from different FFT input sets. This differs from case 2 in that using the remaining data from the 256 bit read requires an increment of the outer loop. This is signalled to the logic by setting ID\_RM\_CONF2.PM to IM (1<sub>B</sub>).

In this case, to avoid discarding data and wasting bandwidth, multiple data blocks can be built in memory by setting the ID\_RM\_CONF2.BLOCKS bitfield to a value other than 0<sub>D</sub>. This bandwidth optimisation is described in [Section 20.3.1.2.6, “Bandwidth Optimisation Integration Processing Mode.” on Page 12](#). This has the effect of collapsing the outer loop as well as the inner loop resulting in a three-dimensional array of input data in the buffer memory.

#### Buffer Memory Switching

If the ID\_RM\_CONF2.PM bit is set to IM, this signals that the data block should contain one dataset for each antenna (i.e. the Inner Loop Offset is set to step across antennae) and the Buffer Memory will be switched when the Inner Loop Repeat Count is reached. Note that in this case, the FFT accelerator will run

<sup>1)</sup> 16 bit precision real data (FORMAT=REAL16BIT) and 16 bit half precision floating point real data (FORMAT=REAL16FP) are not supported with TRNSPS=1.

## Signal Processing Unit 2 (SPU2)

ID\_RM\_CONF2.BLOCKS+1 processing operations for each switch of the buffer memory. Otherwise, the Input Buffer Memory will be switched every four (for 64 bit data) or eight (for 32 bit data) iterations of the Inner Loop or when the Outer Loop Count is updated or all data has been read whichever occurs first.

### 20.3.1.2.5 Bandwidth Optimisation for Default Processing Mode

The memory mapping used to store data after the TC3Ax SPU processes ADC samples means that consecutive addresses in memory contain adjacent samples from the same FFT and that the same FFT result points from different antennae are always separated in the address space. Default mode is designed to work with this memory organisation to optimally construct multiple input datasets from each set of 256 bit reads. Each of the samples in a single data read will be allocated to different input datasets in the buffer memory.

The mode is enabled by setting ID\_RM\_CONF2.TRNSPS=TRN and ID\_RM\_CONF2.PM=DM

To work in default mode either the **ID\_RM\_BLO**.BLO or **ID\_RM\_ILO**.ILO bitfields must be used to iterate across samples in adjacent memory locations.

**Note:** *It is supported that one of the three fields, **ID\_RM\_OLO**.OLO, **ID\_RM\_ILO**.ILO and **ID\_RM\_BLO**.BLO can be set to the sample size as this can be a useful reminder of the expected IDM behaviour. However this is not mandatory. Setting the relevant field to increment by a 256 bit word (32 bytes) will cause the TC3Ax SPU to operate in exactly the same way. Setting the field to an integer multiple of 32 bytes is also supported.*

### 20.3.1.2.6 Bandwidth Optimisation Integration Processing Mode.

Integration Mode Bandwidth Optimisation is intended for the use case where the MATH2 unit is required to integrate results across antennae while the input data manager is using transposed addressing. In this mode, the Input DMA Engine must load data into the Input Buffer Memory so that each constructed data block contains one dataset from each antenna.

The mode is enabled by setting ID\_RM\_CONF2.TRNSPS=TRN and ID\_RM\_CONF2.PM=IM.

In this case, the data block being constructed in memory would use only one sample point from each 256 bit read, as the first dimension of traversing the data cube (Bin Loop Offset) is across FFT results and the second (Inner Loop Offset) is across antennae (i.e. the data blocks constructed are suitable for use by the coherent or non-coherent integration functions of the MATH2 unit). This compares to the default transpose addressing mode, where the first dimension is still across FFT results but the second is across sample points and each read fetches several contiguous sample points which can be used to construct datasets that can be processed as a single data block.

To mitigate the inefficient use of Radar Memory bandwidth, an optimisation submode of Integration Mode controlled by the ID\_RM\_CONF2.BLOCKS bitfield is available.

Enabling this submode allows multiple data blocks to be constructed in the buffer memory. Then, instead of switching the buffer memory between every data block, the base address the Loader Module uses for reads is adjusted instead. When IDM\_RM\_CONF2.PM is set to 1 and a non-zero value is written to ID\_RM\_CONF2.BLOCKS, the Input DMA Engine will try and use some of the excess data read to construct the additional data blocks in the buffer memory.

To support this, the Buffer RAM will be logically sub-divided into multiple segments based on the most significant address bits, one for each data block. The number of data blocks being equal to the value of the ID\_RM\_CONF2.BLOCKS bitfield plus one. BLOCKS can be set to values between  $0_D$ ,  $1_D$ ,  $3_D$ , and  $7_D$ .

The Data Loader Unit will then read each data block sequentially rather than switching the buffer memory and only switch the buffer memory when all the data blocks have been processed. All units downstream of the FFT still process each data block discretely.

Application software must ensure that the data will fit into the sub-divided buffer memory.

## Signal Processing Unit 2 (SPU2)

**Note:** If the number of data blocks built in the buffer memory does not use all of the 256 bits read from Radar Memory, then the same 256 bits will be read on successive passes until all the data is consumed. e.g. if the data in radar memory is 64 bit and BLOCKS is set to  $1_D$ , then the Input DMA will make two passes through the data in Radar Memory. Each pass will create two data blocks in the Buffer Memory.

### 20.3.1.3 Partial-acquisition Counter

The partial-acquisition counter is used to count ramps and hence to determine the absolute position in the measurement cycle. It is reset by software and will then use the configuration information (samples per ramp and number of antennae) valid at the point of reset to count ramps until the next time it is reset by software.

The intended use case of the counter is to allow the TC3Ax SPU to be triggered multiple times during a single measurement cycle while still retaining synchronisation with the input data stream from the Radar Interface(s). This allows the first few ramps of a measurement cycle to be processed to determine useful information about the input data. The TC3Ax SPU can then be reprogrammed to make most effective use of the input data and retriggered at a known ramp count to process the remaining data from the measurement cycle.

The counter can be configured to generate an error or attention interrupt if any particular value is reached with the counter still enabled.

The TC3Ax SPU can be configured to delay start of processing until a particular value is reached.

The operation of functions related to this counter are controlled by the bitfields of the PACTR register.

**Table 785 Partial-acquisition Counter Specific Parameters for ADC Data**

Parameter	Definition	Comments
Initialise the Counter	Reset the value of the partial-acquisition counter to $0_D$ .	<b>PACTR.RST</b>
Enable the Counter	Counter will increment every time the IDM requests a switch of the input buffers.	<b>PACTR.EN</b>
Limit Value for the counter	Comparison value for triggering a pre-programmed event Note that for comparison purposes, the first ramp of a measurement cycle is considered to be ramp 0.	<b>PACTR.LIMIT</b>
Trigger	Trigger TC3Ax SPU data processing on a limit event	<b>PACTR.TRIG</b>
Error	Trigger an error interrupt on a limit event (unless TRIG bit is set and TC3Ax SPU is configured and waiting for data i.e. CTRL register written with MODE=INT)	<b>PACTR.ERR</b>
Attention	Trigger an attention request interrupt on a limit event	<b>PACTR.ATTN</b>
Counter Value	Current state of the Partial Acquisition Counter	<b>PACTR.COUNT</b>

## Signal Processing Unit 2 (SPU2)

Example use case for a measurement cycle with 1024 ramps of 256 samples and 4 antennae with the intention to process the first four ramps, skip two ramps while reprogramming the TC3Ax SPU and then process the remaining 1018 ramps:

- Program TC3Ax SPU for data from RIF, 256 samples and 4 antennae and four ramps, reset the partial-acquisition counter. Set the PACTR Error Interrupt and configure the limit field to 6 (as we expect to have reprogrammed the TC3Ax SPU by the start of the seventh ramp). Trigger the TC3Ax SPU by either internal (CTRL.MODE=INT) or external mode (CTRL.MODE=EXT).
  - This will initialise the Partial-acquisition counter with the correct information to keep track of the ramps
  - The TC3Ax SPU will process 4 ramps and then stop. The TC3Ax SPU DONE interrupt will then trigger the CPU to process the data. The CPU can then take decisions on how to best configure the TC3Ax SPU for the remainder of the measurement cycle
  - The Partial-acquisition counter will continue to track ramps. If the ramp counter reaches six (i.e. the seventh ramp has started) without the PACTR register being updated by the CPU, then the error interrupt will be asserted. This would indicate that the expected reconfiguration time has been exceeded.
- The CPU reprograms the TC3Ax SPU for the remainder of the measurement cycle<sup>1)</sup>. The partial-acquisition counter trigger bit is set and the error bit is set. The CTRL.MODE bitfield must be set to INT or EXT<sup>2)</sup>. The partial-acquisition counter is not reset.
  - The TC3Ax SPU is triggered at the start of ramp seven and the data from the seventh ramp to the one thousand and twenty-fourth ramp is acquired and processed. This trigger must be an internal trigger.
  - Setting the error bit eliminates the race condition of the start of the seventh ramp occurring between the write to the PACTR register and the write to the CTRL register. The error interrupt will be gated if TRIG is set and the CTRL register has been written with MODE=INT or with MODE=EXT and an external trigger has occurred.

The partial-acquisition counter can only be used to monitor input data sourced from the Radar Interface. It cannot be used to track data loaded from Radar Memory.

### Notes

1. *The TC3Ax SPU can process data from the Radar Memory without impacting the operation of the Partial Acquisition Counter once it is configured and running*
2. *Setting the partial acquisition trigger (PACTR.TRIG = 1) without enabling the counter (PACTR.EN = 0) will still enable the trigger logic. Incoming Data will be filtered indefinitely waiting for a limit event which will not occur*

### 20.3.1.4 FFT Data to Antenna Mapping

Some processing units (e.g. the complex windowing function) require that each set of FFT data be associated with an antenna. This is done using the information stored in the ID\_RM\_CONF2.PM and ID\_RM\_CONF2.AM bitfields.

If ID\_RM\_CONF2.PM is set to 1<sub>B</sub>, then the buffer memory datablock will contain one dataset from each antenna. The processing units will therefore use the dataset index to identify the antenna.

If ID\_RM\_CONF2.PM is set to 0<sub>B</sub>, then the data block is assumed to contain data from a single antenna. In this case, the setting of ID\_RM\_CONF2.AM is used to define which of the loop repeat counters (Inner Loop Repeat or Outer Loop Repeat) is used to generate the antenna ID. Alternatively, the antenna ID can be forced to either 0<sub>D</sub> or

---

1) The reconfiguration of the TC3Ax SPU is not expected to change any setting in the ID\_CONF or ID\_CONF2 registers except the ID\_CONF.RAMPS field as the RIF(s) are still running the same measurement cycle and the format of the data input should not have changed. This restriction means that, although BYPASS could be used, reloading bypassed data before the end of the measurement cycle (i.e. in the gaps between partial acquisitions) is not supported.

2) If the partial-acquisition Counter trigger condition is reached before the CTRL register is programmed to enable the TC3Ax SPU, then an error interrupt will always be generated.

## Signal Processing Unit 2 (SPU2)

“dataset index” (equivalent to ID\_RM\_CONF2.PM=1<sub>B</sub>). If the antenna ID is derived from one of the loop counters, only the three LSBs of the counter will be used to keep the antenna ID within the permitted range of 0<sub>D</sub> to 7<sub>D</sub>.

The loop repeat counter value used is chosen at the point the buffer memory is filled and before the counter is incremented for reading the next data, This leads to the following effects:

- If ID\_RM\_CONF2.AM is set to BLR, the antenna ID will always be the three LSBs of the BLR value in the RM\_BLR register.
- If ID\_RM\_CONF2.AM is set to ILR, the antenna ID will be as follows
  - for the final data block before incrementing the Outer Loop Counter, the antenna ID will be the 3 LSBs of the ILR value in RM\_IOLR
  - for other data blocks, the antenna ID will alternate between 3<sub>D</sub> and 7<sub>D</sub> for 32 bit precision complex data and will always be 7<sub>D</sub> for 16 bit precision data (real or complex format) or power data.
- If ID\_RM\_CONF2.AM is set to OLR, the antenna ID will start at 0<sub>D</sub> and will then increment every time the Outer Loop Counter is incremented. If ID\_RM\_CONF2.BLOCKS is not equal to zero, the value will be the final value of the Outer Loop Counter when the IDM completes loading of the buffer memory

### 20.3.1.5 Reading Power Data From Radar Memory

The Input DMA will normally read complex data from the Radar Memory as this is the expected format of FFT results. However, in some cases, further analysis of linear power data stored by a previous processing pass of the TC3Ax SPU may be desirable. This data may be stored as 32 bit unsigned integer or 16 bit IEEE 754 Half Precision Floating Point format and ID\_RM\_CONF2.FORMAT has options for reading these data.

These settings are intended to allow power data to be processed by the MATH2 units and must only be used with the FFT accelerator and MATH1 complex data specific functions bypassed or disabled.

### 20.3.1.6 Reading 16 bit Real Data From Radar Memory

The Input DMA will normally read complex data from the Radar Memory as this is the expected format of FFT results. However, in some case, memory restrictions may require data to be stored as 16 bit precision, real only data. Reading 16 bit precision, real data can be done by setting ID\_RM\_CONF2.FORMAT to REAL16BIT for integer data or REAL16FP for half precision floating point. If either of these options is used, then address transpose mode is not available and ID\_RM\_CONF2.TRNSPS must be set to 0<sub>B</sub>.

#### 20.3.1.6.1 Buffer Memory Switching for 16 bit real data

If the ID\_RM\_CONF2.PM bit is set to IM, this signals that the data block should contain one dataset for each antenna (i.e. the Inner Loop Offset is set to step across antennae) and the Buffer Memory will be switched when the Inner Loop Repeat Count is reached. Otherwise, the Input Buffer Memory will be switched every eight iterations of the Inner Loop or when all data has been read whichever occurs first.

### 20.3.1.7 Reading Half Precision Floating Point Data From Radar Memory

For half precision floating point, denormalised numbers, Infinities and NaNs should be avoided in the input data. Denormalised numbers will be treated as zero. Infinities and NaNs will be set to either maximum negative or maximum positive values depending on the sign bit to allow processing to continue and will not propagate. The TC3Ax SPU has no capability to process signalling NaNs.

### 20.3.1.8 Reading Compressed Complex Data From Radar Memory

Data written to Radar Memory by the TC3Ax SPU in compressed format using ODP\_CONF.COMPRESS==1 may be read by the TC3Ax SPU by setting RM\_CONF2.FORMAT to COMPRESS. The data is unpacked such that each bin

## Signal Processing Unit 2 (SPU2)

uses the same number of bits as 16-bit complex data in the Buffer Memory. The Data Loader Unit converts the unpacked data into 32-bit complex. See “[Data Loader Unit](#)” on Page 17 for details.

### 20.3.1.9 Data Storage in Buffer Memory

The datasets will be stored linearly in the buffer memory in the order they should be processed. This allows the Data Loader module to progress sequentially through the buffer memory address space.

### 20.3.2 Streaming Processor 1

Streaming Processor 1 retrieves data from the buffer memory loaded by the Input DMA Engine and processes the information for use by the FFT accelerator

#### 20.3.2.1 Double Pass Mode

The TC3Ax SPU allows all processing operations to be run twice per data block. This mode is enabled using the DPASS\_CONF.EN bitfield. Once the data has been loaded to Buffer RAM by the Input DMA Engine, the subsequent modules in the processing pipeline have the option to

- read the data twice and apply different processing options. This is the default “double pass” mode
- independently apply the different parameter sets defined for the window function to successive sequences of data blocks. See [Section 20.3.2.1.2](#) below.

This results in two sets of results being written to Radar Memory.

**Table 786 Parameters Specific to Double Pass Mode**

Parameter	Definition	Comments
Enable Mode	Configure TC3Ax SPU to process each data block twice.	<b>DPASS_CONF.EN</b>
Switch Mode	Configure TC3Ax SPU to process successive data blocks with different parameter sets. Number of data blocks between parameter switches configured by DPASS_CONF.COUNT	<b>DPASS_CONF.SWITCH</b>
Window Parameter Switch Count	Switch the window parameters when this many data blocks have been processed	<b>DPASS_CONF.COUNT</b>
Window Parameter Switch Count Enable	Enable the Window Parameter switch count	<b>DPASS_CONF.EN_CNT</b>

The values to be used in the second processing pass are stored in the BE1\_<REGNAME> registers.

The following register fields should not be changed between processing passes. A write to these fields in either copy of the register will be reflected in both registers.

## Signal Processing Unit 2 (SPU2)

**Table 787 Mirrored Register Fields**

Register	Field
LDR_CONF	DRPL
	DRPF
	EXPNT
	SIZE
	FFTBYPS
	IFFT
LDR_CONF2	PADF
	PADDIS
UNLDR_CONF	FORMAT
	EXPNT

**Note:** Note that the mode of the FFT module (FFT or IFFT) cannot be varied between passes so the LDR\_CONF.IFFT bit is mirrored.

### 20.3.2.1.1 Double Pass Switch Mode

The behaviour of Double Pass Mode can be modified by setting the DPASS\_CONF.SWITCH bitfield. In this case, each data block is only processed once and the different parameter settings are applied to consecutive data blocks.

### 20.3.2.1.2 Window Parameter Switch

The switch of the two possible sets of parameters for the window function can be made independent of the double pass functionality by setting the DPASS\_CONF.EN\_CNT. When this bit is set, the window parameters are selected independently of the double pass function and applied to sequences of data blocks. The number of data blocks per sequence is configured by the DPASS\_CONF.COUNT bitfield and is set to COUNT+1. If COUNT=0, then the different parameter settings are applied to alternate data blocks. If COUNT=7, then the different parameter settings are applied to sequences of 8 data blocks.

### 20.3.2.2 Data Loader Unit

The Data Loader will read data from memory and reformat it before pushing it into the MATH0 and MATH1 unit pipelines.

#### 20.3.2.2.1 Data Reformatting for MATH0

As only ADC data is sent to MATH0, the only reformatting is conversion from unsigned to 2's complement format when the Input DMA configuration has ID\_CONF2.SIGNED == UNSIGNED. In this case the DC Offset Removal function handles the reformatting (see “DC Offset Removal” on Page 21 for details). When the input is complex this conversion is done separately for both the real and imaginary components.

#### 20.3.2.2.2 Data Reformatting for MATH1

The format of the input data can reformat input data by resizing it from 16 to 32 bit precision and by converting it from real to complex format. Operations will occur in the following order

## Signal Processing Unit 2 (SPU2)

- 16 to 32 bit precision
- real to complex conversion

**Table 788 Loader Configuration Parameters**

Parameter	Definition	Comments
Number of samples	Number of samples per FFT currently in the buffer RAM	Passed from the Input DMA Engine
Number of data sets	Number of FFTs to run	Passed from the Input DMA Engine
Data precision	16 or 32 bits	Passed from the Input DMA Engine
Data type	Real or complex	Passed from the Input DMA Engine
Data format	Integer, floating point or compressed	Applicable to Radar Memory data only. Passed from the Input DMA Engine
Integration Mode Bandwidth Optimisation	Buffer Memory Configuration. Number of Datablocks in the buffer memory	Passed from the Input DMA Engine
Signed or Unsigned	Input Data is unsigned or 2's complement format	Applicable to ADC data only, passed from the Input DMA Engine

### Data Reformatting

The format of the data in the buffer memory is known as it is programmed into the Input DMA Engine configuration. All data to be processed by the FFT accelerator must be in 32 bit complex format.<sup>1)</sup> Real data is converted to complex data with a zero imaginary component. 16 bit data is extended to 32 bit using the method specified by the LDR\_CONF.EXPNT bitfield. This controls how the 16 bit data is justified within the 32 bit output field. The type of data (signed or unsigned) is passed from the Input DMA Engine and is derived from the ID\_CONF2.SIGNED bitfield. This determines whether a signed or unsigned extension to 32 bits is used. When MATH0 is enabled and not bypassed the data is always sign extended because the DC Offset Removal function converts to signed data.

When using data from Radar Memory, LDR\_CONF.EXPNT has a maximum permissible value of  $16_D$  and aligns 16 bit fixed exponent format data. When using data from the Radar Interface, the number of significant bits can be less than  $16_D$  so the maximum permissible value of LDR\_CONF.EXPNT can be increased to allow for 10 bit data right aligned in the 16 bit field.

If the shift operation causes a loss of significant data bits, then the output of the shift operation will saturate at full scale.

If the input data format is complex, then both real and imaginary components are treated in the same manner.

If the input data format is compressed (ID\_RM\_CONF2.FORMAT=COMPRESS) then the LDR\_CONF.EXPNT field has no effect. The unpacked compressed data in the Buffer RAM is expanded to 32-bit complex data.

### Integration Mode Bandwidth Optimisation

This is described in [Chapter 20.3.1.2.6, “Bandwidth Optimisation Integration Processing Mode.” on Page 12.](#)

1) The FFT accelerator itself operates on 24-bit data internally but the rest of the FFT data path is 32-bits.

## Signal Processing Unit 2 (SPU2)

The Integration Mode Bandwidth Optimisation of the Input DMA Engine results in multiple datablocks being present in the buffer memory. The Buffer memory would normally be switched between datablocks. When this mode is enabled, instead of switching the buffer memory, the buffer memory address is offset to point to the next datablock. The MATH1, FFT and all units downstream of this point therefore see separate datablocks as normal.

### 20.3.2.3 MATH0 Unit

The MATH0 unit handles processing of ADC data<sup>1)</sup> before it is passed to the MATH1 unit. It has time-domain interference detection and removal functions as well as a DC offset removal function. The interference is assumed to occur in one or more bursts within a ramp. The unit may be bypassed by setting M0CTRL.BYPASS bitfield. When bypassed, the unit may still process data, writing it to Radar Memory. It may also detect interference bursts, recording their locations in registers without writing any data to Memory. The MATH0 unit processes data on a ramp-by-ramp and antenna-by-antenna basis as it comes out of the Buffer RAM. This means, at any one stage in the processing, only samples from one antenna are available in the unit.

The MATH0 unit has a double buffer with two banks, called the Ramp Buffer, and each bank has capacity to hold two copies of a whole ramp from one antenna. The buffer has two write channels and two read channels with one read and one write per bank. Data flows along two paths in MATH0: forwards and backwards as shown in “[MATH0 Unit Architecture](#)” on Page 20. This enables the signal to be filtered in both directions from the start and end of each interference burst.

When MATH0 is active data is read from the Buffer RAM starting with the last sample in a ramp in reverse order. Otherwise data is read starting with the first sample in natural order. The DC Offset and Transient Removal functions operate on a sample-by-sample basis and have no state so they not affected by the order of the data. One copy of data is loaded into Ramp Buffer after optional processing by these units. The same data is also fed to the Detector and Backwards Filter and the filtered output is written into Ramp Buffer.

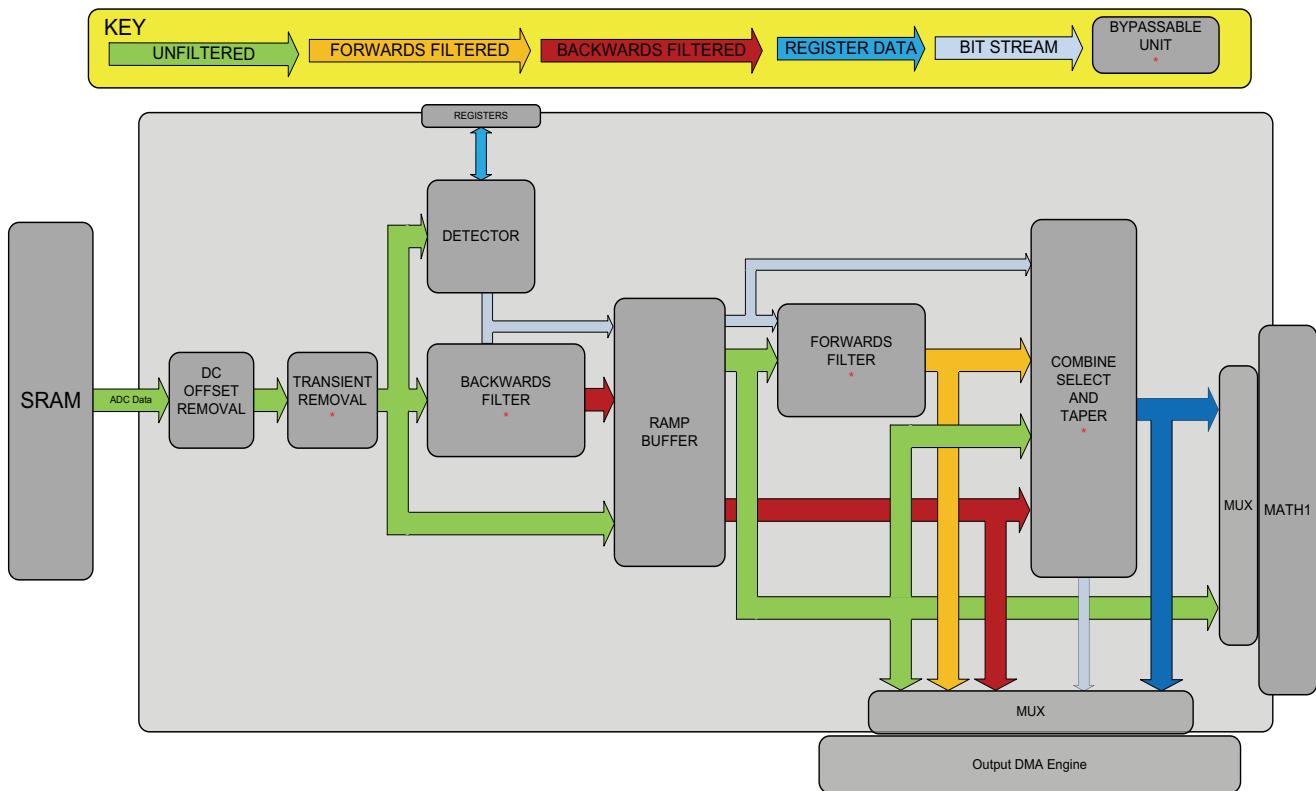
The data is read out of the Ramp Buffer in two streams, filtered and unfiltered such that the data samples are in their original order and aligned to each other. The unfiltered data is filtered by the Forwards Filter. The forwards and backwards filtered data are fed along with the unfiltered data to the Combine Select and Taper (CST) unit.

The Detector unit operates on backwards data so that it is able to control the backwards filter directly. The location of any detected interference is recorded in a set of registers and presented as a bit stream to the filter and the Ramp Buffer. The Detector results are then read in the correct order to be used by the subsequent units that operate in a forwards direction.

All of the units marked with a star (\*) in [Figure 258 “MATH0 Unit Architecture” on Page 20](#) are bypassable. When MATH0 is bypassed by setting M0CTRL.BYPASS data will still be processed by the DC Offset and Transient Removal units if they are enabled. MATH0 functions are not supported when ID\_CONF2.SRC==RM and the unit should be disabled by clearing M0CTRL.EN.

<sup>1)</sup> MATH0 can only process data when ID\_CONF2.SRC==RIF or BYPASS. Enabling MATH0 when ID\_CONF2.SRC==RM MATH0 is not supported.

## Signal Processing Unit 2 (SPU2)



**Figure 258 MATH0 Unit Architecture**

**Table 789 MATH0 General Configuration**

Parameter	Definition	Comments
Operating mode	Select the type of data being processed: real or complex	<b>ID_CONF2.FORMAT</b> Controlled by the IDM FORMAT
Operating mode	Select the type of data being processed: unsigned or signed	<b>ID_CONF2.SIGNED</b> Controlled by the IDM FORMAT NB This setting only affects sign/zero extension for DC offset removal
Enable	Enable the unit.	<b>MOCTRL.EN</b> Enabling MATH0 with ID_CONF2.SRC==RM is not supported
Bypass	Bypass MATH0 and send data directly to MATH1 after DC Offset and Transient Removal. Data is still processed by MATH0 and available for writing to Radar Memory	<b>MOCTRL.BYPASS</b>
Output base address	Address in the Radar Memory where MATH0 will write data if enabled by RPLCTRL.OSEL	<b>MOCTRL.BASE</b>

## Signal Processing Unit 2 (SPU2)

**Table 789 MATH0 General Configuration (cont'd)**

Parameter	Definition	Comments
Output data enable and selection	Selects what data is written to Radar Memory. See “ <a href="#">Output to Radar Memory</a> ” on Page 27 for details.	RPLCTRL.OSEL
Number of samples per ramp	The number of samples per ramp is defined by the Input DMA configuration. When the MATH0 unit is enabled, the minimum number of samples allowed is 128.	ID_CONF.SMPLCNT

### 20.3.2.3.1 DC Offset Removal

This unit subtracts the mean value, calculated over a complete ramp, from each sample giving a resulting data stream that is free of any DC offset. The mean value is calculated as the data is written to the Buffer RAM so that it is available when the first sample is read from the Buffer RAM. For complex data, a separate mean is calculated for the real and imaginary components and these are subtracted from the relevant component in each sample.

When ID\_CONF2.SIGNED=UNSIGNED the data input to DC Offset Removal is zero extended to 17 bits to convert to a signed value without loss of precision. The output from DC offset is saturated to a signed 16-bit value. The rest of MATH0 operates on signed ADC data. The calculation of the DC offset shall be performed to a precision of plus or minus one LSB of the 16-bit value provided the correct reciprocal value is configured in the SCRECIP register.

The operation of Data Reformatting is affected when MATH1 receives data from MATH0 (see “[Data Reformatting for MATH1](#)” on Page 17 for details).

**Table 790 DC Offset Removal Configuration**

Parameter	Definition	Comments
Enable	DC offset removal is always enabled if MATH0 is enabled	M0CTRL.EN
Sample Count Reciprocal	DC offset calculates the mean value using a reciprocal of the number of samples programmed in this register	SCRECIP.EXP SCRECIP.MANT

### 20.3.2.3.2 Transient Removal

This unit compares each sample magnitude with a threshold magnitude and replaces any sample that exceeds that threshold with a new value depending on the mode controlled by M0CTRL.TRRM. The function of this unit is described by the following equation:

(20.1)

$$f(x) = \begin{cases} \text{VALUE, if } |x| > \text{THRESH} \\ x, \text{ otherwise} \end{cases}$$

where VALUE is either zero or  $\text{sgn}(x) \times \text{THRESH}$  depending on the value of M0CTRL.TRRM. Also the  $\text{sgn}()$  function takes the sign of its argument.

## Signal Processing Unit 2 (SPU2)

**Table 791 Transient Removal Configuration**

Parameter	Definition	Comments
Enable	Remove transients if set otherwise pass input unchanged	<b>M0CTRL.TREN</b>
Mode	When a sample exceeds the threshold, set to zero value (ZERO) or the threshold value (THRESHOLD)	<b>M0CTRL.TRRM</b>
Threshold	The magnitude threshold value to compare with the magnitude of the sample value	<b>TRTHRESH.THRESH</b>

### 20.3.2.3.3 Interference Detection

The Interference Detector function identifies any parts of the ADC input stream where the difference in power level from one sample to the next exceeds a configurable threshold, DETCFG.THRESH. It includes an optional high-pass filter before linear power and difference calculation. The power difference is compared with the configurable threshold and the result is feed to a delay line with a configurable number of bits. Each bit that is a one represents a sample where the power difference threshold is exceeded. The number of bits that are one in the delay line is counted and this population count is compared to another configurable threshold, DETCTRL.PCTHRESH.

The Detector operates on sample data stream read from the Buffer Memory so that it may control the backwards filter. The output from the Detector is also used by the forwards filter and the Combine Select and Taper (CST) unit. The detection information is re-ordered as necessary before being passed to these units or being output to Memory.

The Detector may be configured to write the detection results to Memory using DETCTRL.OE. The output consists of a stream of bits, one per input sample in the forwards direction. A given bit is set if interference has been detected according to the algorithm, otherwise the bit shall be zero. The bit stream is packed into 256 bit words before being written to memory. Any remaining data at the end of a measurement cycle shall be padded with zeros to 256 and flushed to Memory. When double pass is active two sets of data shall be written to Radar Memory.

Any interference bursts that are detected are also recorded in a set of status registers MDM0\_CNT and MDM0\_BURSTb. This data is also stored in order in a forwards direction. These registers are written by hardware and may only be read by software. The CST unit also uses this data to locate any bursts. If there are more than  $16_D$  bursts in a ramp, only the last  $16_D$  will be seen by the unit. A flag is set in MDM0\_CNT when the CST unit gets inconsistent burst data.

The high-pass filter is a discrete realization of a first order RC filter where each output  $y$  is given by:

(20.2)

$$y_i = \alpha \times (y_{i-1} + x_i - x_{i-1})$$

Where  $x_i$  and  $x_{i-1}$  are the current and previous input samples. The real fractional coefficient, alpha, is configured by an 8-bit unsigned bit-field called DETCTRL.ALPHA. The output of the filter is saturated to 16 bits. The feedback term  $y_{i-1}$ , begin the previous filter output, is represented as a signed 18-bit value. The 8 LSBs of the product are

## Signal Processing Unit 2 (SPU2)

dropped and the result of the sum, inside the parentheses are saturated to 18 bits. When in complex mode each component, real and imaginary, is filtered separately.

### Detector Configuration and Status

**Table 792 Interference Detection Configuration**

Parameter	Definition	Comments
Enable	Enable the detector	<b>DETCTRL.EN</b>
Output enable	If set when EN set, write a bit mask to Radar Memory starting at the location defined in DETBASE.BASE.  Note: <i>When EN is not set this bit is ignored.</i>	<b>DETCTRL.OE</b>
High-pass filter enable	When set to OFF, the filter will be bypassed.	<b>DETCTRL.HPFEN</b>
High-pass filter coefficient	Defines the cut-off frequency of the filter. Setting to zero will disable the detector function.	<b>DETCTRL.ALPHA</b>
Detection power difference threshold	When the power difference is greater than this value a one is feed into the detection window delay line.	<b>DETCFG.THRESH</b>
Detection window size	The number of samples either-side of the current sample. The total number of samples in the window is 1+2xWINSIZE.	<b>DETCTRL.WINSIZE</b>
Population count threshold	The threshold applied to the population count. When set to zero only one sample needs to exceed the power difference threshold.	<b>DETCTRL.PCTHRESH</b>

**Table 793 Detector Status**

Parameter	Definition	Comments
Burst count	Incremented each time an interference burst is detected and a MDM0_BURSTb register is set by the hardware. Reset at the start of a measurement cycle.	<b>MDM0_CNT.CNT</b>
Burst count overflow	Set if the count overflows. Stays set and only gets reset when the SPU is triggered.	<b>MDM0_CNT.OV</b>

## Signal Processing Unit 2 (SPU2)

**Table 793 Detector Status (cont'd)**

Parameter	Definition	Comments
Excess bursts detected	Set if CST gets inconsistent burst data. Stays set and only gets reset when the SPU is triggered.	<b>MDM0_CNT.XS</b>
Burst descriptor	A set of sixteen registers that describe a burst by its ramp number, location in that ramp and its length. The START and LENGTH fields are set according to the detection on first antenna/channel of the ramp number given in the NUM field. A zero value of LENGTH equates to a burst with one sample in it.	<b>MDM0_BURSTb (b=0-15)</b>

**Note:** *The detector burst descriptor metadata is collected on a ramp-by-ramp basis because it is assumed that any interference will affect all antennae in a similar way. Given that the ADC sample rate is of the order of one thousandth of the Radar frequency this assumption can be seen to hold for all use cases.*

### 20.3.2.3.4 Ramp Buffer

The ramp buffer allows each ramp to be filtered from the beginning forwards and from the end backwards for each detected burst. The data is read once from the Buffer RAM and stored twice in the ramp buffer, once before backwards filtering and once after. The filter buffer has two banks so that a second ramp may be written while the first is being read. This double buffering is necessary because the whole ramp must be stored to be able to access it in both directions.

Each ramp buffer memory bank is large enough to hold two complete 2048 sample ramps where each sample is a 16 bit real value. For complex ADC data, the maximum number of samples that can be stored per ramp is halved.

### 20.3.2.3.5 Filtering

There are two identical filter units both configured by the same set of registers, with the exception of the FILTCTRL.CONJ bitfield. Each implements an Infinite Impulse Response (IIR) filter which includes a recursive section and a non-recursive section. The only difference between the two units is that each operates on data samples in opposite orders, forwards and backwards. The input and output data are treated as signed fixed point data with one integer bit and 15 fractional bits. The coefficients are treated as signed fixed point data with three integer bits and 13 fractional bits. The output of each filter is saturated to the minimum and maximum representable values while the internal feedback path for the recursive section is not saturated.

The filters are enabled by FILTCTRL.EN. When not enabled they are bypassed and data flows directly to the next stage: the Ramp Buffer for the backwards filter and the Combine Select and Taper unit for the forwards filter. When the filters are enabled and there is no interference detected the filters are transparent with their output being equal to their input but they maintain their state by shifting the input sample values into the delay line.

The filter unit has a configurable delay line allowing the taps to be allocated between a recursive section and non-recursive section. The non-recursive section holds delayed input samples and the recursive section holds delayed output samples. The delay line has a total of 16 16-bit elements. These elements may be allocated to the recursive section by setting the number of recursive coefficients in FILTCTRL.NCOEFR. The remaining 16 - NCOEFR coefficients are available to the non-recursive section. The number of non-recursive coefficients that are active is defined by FILTCTRL.NCOEFNR. By setting FILTCTRL.NCOEFR to zero, the filter becomes entirely non-

## Signal Processing Unit 2 (SPU2)

recursive, in other words, it shall be a Finite Impulse Response filter (FIR). By setting FILTCTRL.NCOEFR to zero, the filter becomes a resonator as the input signal has no effect on the output. Setting both FILTCTRL.NCOEFR and FILTCTRL.NCOEFNR to zero shall result in the output from the filter being zero always.

When ID\_CONF2.FORMAT==COMPLEX and the unit is operating on complex data, two delay elements are used for each sample and the coefficients are used pairs. In this mode the LSB of NCOEFR shall be ignored and the number of complex coefficients shall be the value in the MSBs of that field.

The values in the FILTCOEFc registers shall be used in the recursive section starting with the lowest numbered register field first. So, FILTCOEF0.C0 shall be the factor used to multiply the filter output delayed by one sample, FILTCOEF0.C1 for the filter output delayed by two samples and so on. The coefficient values shall be used in the non-recursive section starting with the highest numbered register field first. So FILTCOEF8.C1 shall be the factor used to multiply the filter input, FILTCOEF8.C0 shall be the factor for the filter input delayed by one sample and so on.

When ID\_CONF2.FORMAT==COMPLEX one complex coefficient is held in each register with C0 holding the real part and C1 holding the imaginary part.

The initial filter state at the start of a burst is set to the last N input samples. The samples are loaded into the filter delay lines and moved along as if the filter were operating normally except that, when no interference is detected, the filter output is kept equal to its input. The value of N in this case is notional given the number of taps in the relevant filter section, recursive or non-recursive, may be different to each other. Each section of the filter shall be initialised with the appropriate number of data samples. If an interference burst occurs at the start or end of a ramp and less than N samples have been loaded into the filters, the un-initialized filter taps shall have zero values.

**Table 794 Filter Configuration**

Parameter	Definition	Comments
Enable	Enable filtering, bypass	FILTCTRL.EN
Number of recursive coefficients	Allocate this many of the available coefficients to the recursive section. The rest are available to the non-recursive section.	FILTCTRL.NCOEFR
Number of non-recursive coefficients	Allocate this many of the available coefficients to the non-recursive section. NCOEFR takes priority so the maximum value of non-recursive real coefficients can never be more than 16 - NCOEFR (8 - NCOEFR for complex)	FILTCTRL.NCOEFNR
Coefficient conjugation	Enable conjugation of complex coefficients in the backwards filter. Has no effect on the forwards filter. Has no effect when ID_CONF2.FORMAT==REAL	FILTCTRL.CONJ
Filter coefficients	A set of 8 registers containing 16 16-bit values used for coefficients.	FILTCOEFc (c=0-7)

### 20.3.2.3.6 Combine Select and Taper

This Combine Select and Taper (CST) unit has three sample data inputs and one control input from the Detector unit. It outputs data to MATH1 and, optionally, to Radar Memory. The unit has two operating modes called SOFT

## Signal Processing Unit 2 (SPU2)

and HARD, selected by the RPLCTRL.MODE field. In SOFT mode the unit is controlled by a set of BURSTb registers while in HARD mode the Detector output is the control. For each sample the unit selects between the unfiltered input and a replacement sample value. The replacement value is control by the RPLCTRL.SRC field and may be one of the following:

- ZERO - if tapering is enabled the last sample before the burst is held at the filter inputs otherwise the combined output of CST unit shall be zero for all samples in a burst
- FILTER - the mean of the forward and backward filter outputs with optional tapering
- FORWARD - the output of the Forward filter

**Table 795 Combine Select and Taper Configuration**

Parameter	Definition	Comments
Enable	Enable replacement of samples.	RPLCTRL.EN
Source	Select the source of replacement samples.	RPLCTRL.SRC
Mode	Select what determines the replacement. SOFT mode uses BURSTb registers, HARD uses the output of the Detector	RPLCTRL.MODE SOFT or HARD
Number of Bursts	Used when MODE==SOFT, defines how many bursts are to be replaced	RPLCTRL.NBURSTS
Burst Descriptors	Used when MODE==SOFT. Each register describes a burst in terms of the ramp number (NUM) and sample number (START) and its length (LENGTH). The burst length is LENGTH+1 so the minimum length is 1.	BURSTb These registers are programmed by software. The MDM0_BURSTb registers are written by the Detector Function.
Tapering enable	Enable tapering of the replacement samples.	RPLCTRL.TPREN

### Soft Mode

When RPLCTRL.MODE==SOFT the CST function uses the BURSTb registers to determine where to replace samples. The fields of BURSTb are defined in the same way as the metadata MDM0\_BURSTb. The RPLCTRL.NBURSTS field defines how many of the BURSTb registers shall be used. The BURSTb registers shall be used in numerical order from BURST0 to BURST{NBURSTS}. If NBURSTS==0<sub>D</sub> then no samples shall be replaced. Longer burst lengths may be programmed than would be possible with the LENGTH field size of 10 because abutting or overlapping bursts shall be treated as a single burst.

### Tapering when FILTER is Replacement Source

The tapering function for the filter source is calculated by finding the 16-bit reciprocal of the burst length (DELTA) and using this to decrement a tapering value  $t_i$  that starts at 1.0, represented as an unsigned 16 bit value FFFF<sub>H</sub>. When the burst length is one, and the LENGTH field of the burst descriptor is zero DELTA shall be zero. When the burst length is two DELTA shall be FFFF<sub>H</sub>. For all other burst lengths the DELTA value shall be calculated such that the tapering value always reaches zero, with a tolerance of 1 LSB, by the last sample in the burst. The tapering value must never wrap past zero. The tapering value shall decrease monotonically over the whole burst and be correct to within plus or minus one LSB.

## Signal Processing Unit 2 (SPU2)

The tapering operation used for the filter replacement source (SRC==FILTER) is calculated by first subtracting the forwards filter output sample from the backwards filter sample to yield a 17 bit value. This is multiplied by the current 16-bit unsigned taper function value producing a signed 33 bit value. That maximum negative value this product can take is  $\text{FFFF}_{\text{H}}$  because the maximal operands are  $\text{FFFF}_{\text{H}}$  and  $10001_{\text{H}}$  (max negative - max positive). The signed 16 bit value taken from bits 31:15 of this result is then subtracted from the signed 17-bit value of the sign-extended current backward filter output. The 17-bit result of this subtraction operation can never overflow 16 bits because the calculation always gives a value in the inclusive range between the values of the forwards and backwards filter outputs. The output is taken from bits 15:0 of the subtraction result. This calculation is shown the following equation:

(20.3)

$$\text{out}_i = \text{backward}_i - t_i \times (\text{backward}_i - \text{forward}_i)$$

If RPLCTRL.TPREN is not set when RPLCTRL.SRC==FILTER then the output of the tapering unit shall be the mean of the forwards and backwards filter outputs. This is calculated by adding the sample values and shifting right by one place, truncating to 16 bits.

### Tapering when ZERO is Replacement Source

When tapering is enabled the inputs to the filters are held at the last sample value before the burst for the duration of an interference burst. The combined output shall be calculated in the same way as when RPLCTRL.SRC==FILTER and tapering is disabled giving the mean of the two filter outputs.

When tapering is disabled the samples between START and START+LENGTH-1 of a burst shall be all replaced with zero in the combined output.

### Bursts at the Start or End of a Ramp

When a burst is at the start or end of a ramp there is no data before or after the burst to use in the replacement functions. In these cases zero sample values shall be used in place of the missing data.

#### 20.3.2.3.7 Output to Radar Memory

The MATH0 unit has two output channels, one for the detector bit stream and the other for sample data.

The detector bit stream is only available as an output if the CST function is in HARD mode. The bit stream output is packed into 256-bit words with any unused bits in the last word of a measurement cycle being set to zero.

The sample data output may come from one of the following sources in the MATH0 unit:

- The output of the Combine Select and Taper unit
- The output of the filters - both filter outputs are interleaved
- The output of the Transient Removal unit - this output is still available if the unit is bypassed

The format of this output data is defined by ID\_CONF2.FORMAT being either 16-bit real or 16-bit complex and shall be capable of being read by the IDM with suitable parameter settings. Specifically, data shall be padded with zeros to fill the last 256-bit word at the end of each FFT data set to ensure the next data starts at a 256-bit word boundary.

## Signal Processing Unit 2 (SPU2)

**Table 796 Output Configuration**

Parameter	Definition	Comments
MATH0 Output select	Select which, if any, data is output for Radar Memory	<b>RPLCTRL.OSEL</b>
MATH0 Output Base Address	Base address of where in Radar Memory data shall be written	<b>M0CTRL.BASE</b>
Detector Output enable	Enable output of detector bit stream to Memory	<b>DETCTRL.OE</b>
Detector Output Base Address	Base address of where in Radar Memory data shall be written	<b>DETBASE.BASE</b>

### 20.3.2.4 MATH1 Unit

The MATH1 unit handles transforms applied to the data before the FFT processor. The operations will be applied in the following order:

- Zero Insertion
- Truncation
- Windowing
- Phase Shift
- Padding

**Note:** All of these operations except Zero Insertion are available even if the FFT Engine is bypassed by setting the LDR\_CONF.FFTBYP bit.

**Table 797 MATH1 Configuration Parameters**

Parameter	Definition	Comments
Number of samples	Number of samples per FFT to be loaded into the FFT accelerator	<b>LDR_CONF.SIZE</b>
Number of data sets	Number of FFTs to run	passed from the Loader Unit
Zero insertion enable	Enable zero insertion function Function not available when LDR_CONF.FFTBYP is set.	<b>LDR_CONF2.ZI</b>
Zero insertion mask	A set of registers defining a mask of up to 256 bits	<b>ZI_MASKn</b>
Window base address	Base address in Config Memory where the windowing functions are stored	<b>LDR_CONF2.WBASE</b>
Antenna window offset address	Offset from the Window base address of the window function for a particular antenna	<b>Ax_ANTFOST.ADROFSTy</b> <b>where x=0-3,y=0-1</b>
Padding activation	Padding enable / disable	implicit activation when the FFT size from the loader does not match the FFT size for the FFT accelerator

## Signal Processing Unit 2 (SPU2)

**Table 797 MATH1 Configuration Parameters (cont'd)**

Parameter	Definition	Comments
Leading padding	Position of the 1st operand in the FFT (others being zeroed).	<b>LDR_CONF2.PADF</b>
Padding disable	Disable padding if LDR_CONF.FFTBYPSET	<b>LDR_CONF2.PADDIS</b>
Fast phase shift	Does a 0/90/180/270 phase shift without using the windowing memory	<b>LDR_CONF.PHSHFT</b>

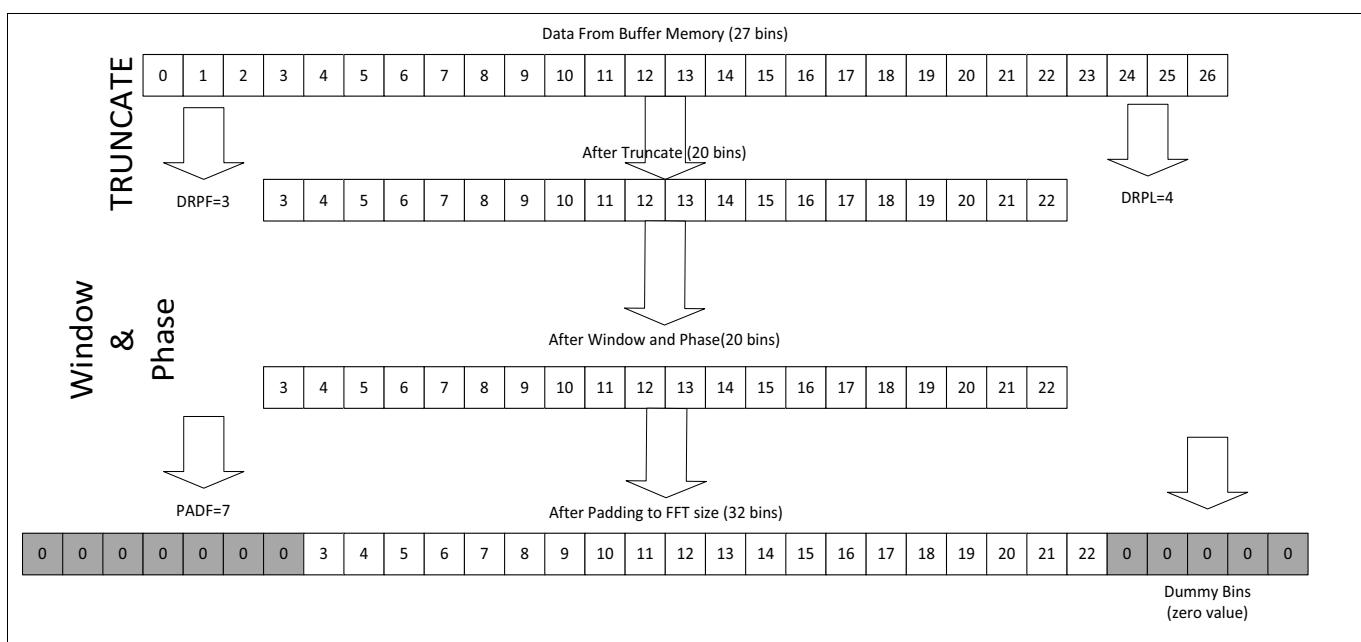
### 20.3.2.4.1 Zero Insertion

The Zero Insertion function is enabled by the LDR\_CONF2.ZI bit. When enabled no truncation nor leading padding shall be performed (LDR\_CONF2.PADF shall be ignored). The input data is read from the Buffer Memory according to the bit mask held in the ZI\_MASKn registers and the FFT size defined by LDR\_CONF.SIZE. For each integer K from zero up to LDR\_CONF.SIZE-1 the corresponding bit b in ZI\_MASKn, where n = K DIV 32 and b = K MOD 32, is tested. If the bit is 1<sub>B</sub>, the next data sample is read from the Buffer Memory and passed to the windowing function, otherwise a zero value is passed. If the number of ones in the bit mask is not equal to the number of samples defined by the Input DMA Engine configuration, the behaviour of Zero Insertion is not defined. If LDR\_CONF.SIZE is more than 5<sub>H</sub>, meaning the FFT size is more than 256 samples, then the FFT input samples beyond the 256th shall be set to zero when Zero Insertion is enabled.

*Note:* The Zero Insertion function is only supported when ID\_CONF2.SRC==RM

### 20.3.2.4.2 Overview of Data Truncation and Padding

The diagram below gives a functional overview of how the operations described below affect the bins of the data loaded from the buffer memory



**Figure 259 MATH1 Operation Overview**

*Note:* No truncation or padding is done if Zero Insertion is enabled.

## Signal Processing Unit 2 (SPU2)

### 20.3.2.4.3 Truncation

Truncation is performed by two register fields, LDR\_CONF.DRPF and LDR\_CONF.DRPL. The reference for the dataset size is taken from the Input DMA Engine Configuration, ID\_CONF.SMPLCNT field. DRPF is used to truncate from the beginning of the dataset and DRPL to truncate from the end of the dataset.

### 20.3.2.4.4 Windowing

Windowing is enable using the LDR\_CONF2.WINEN bit. The coefficients for the windowing function are stored in the configuration memory. The output of the windowing unit is scaled so that the maximum value of the window coefficients is  $+(2^{31}-1)/2^{31}$  and the minimum value is -1.

To retain precision, any needed truncation will be done at the output of arithmetic operations rather than the input. Where a result needs to be scaled down to fit into an output, the result will be rounded down to nearest integer after division.

The base address of the window coefficients is stored in LDR\_CONF2.WBASE. Additionally an address offset can be defined for each antenna. The offset is defined in bytes. If the address offsets are set to zero, the same window will be used for all antennae. Offsets are stored in the ANTOFST fields of the A[0|1]\_ANTOFST registers in the following order:

**Table 798 Antenna to Offset Register Mapping**

Antenna	Register	Bitfield
0	A0_ANTOFST	ADROFST0_ANT
1	A0_ANTOFST	ADROFST1_ANT
2	A1_ANTOFST	ADROFST0_ANT
3	A1_ANTOFST	ADROFST1_ANT
4	A2_ANTOFST	ADROFST0_ANT
5	A2_ANTOFST	ADROFST1_ANT
6	A3_ANTOFST	ADROFST0_ANT
7	A3_ANTOFST	ADROFST1_ANT

Antenna number selection is controlled by the ID\_RM\_CONF.AM and ID\_RM\_CONF.PM bitfields.

The window size must be the same as the truncated data set (i.e. before the padding operation).

The window data can be stored in several formats to save memory. Maximum precision is achieved using 32 bit complex window coefficients. However the data can also be stored in 32 bit real format, in which case the windowing function can affect magnitude but not phase. Further compression can be achieved by using 16 bit formats. Format is set using the LDR\_CONF.FRMT bitfield.

Window coefficients stored as 16 bit are extended to 32 bit by extending with 16 LSBs set to  $0_B$  so that the stored coefficient is left justified in the 32 bit value.

### 20.3.2.4.5 Phase Shift

The phase shift is achieved by sign manipulation of the real and imaginary components of the complex data values as described in the following table.

## Signal Processing Unit 2 (SPU2)

**Table 799 Phase Shift Operation**

Shift (degrees)	LDR_CONF.PHSHFT	Output to Input Mapping	
		Real OP	Imaginary OP
0	00	(Real IP) * 1	(Imag IP) * 1
90	01	(Imag IP) * -1	(Real IP) * 1
180	10	(Real IP) * -1	(Imag IP) * -1
270	11	(Imag IP) * 1	(Real IP) * -1

### 20.3.2.4.6 Padding

The truncated dataset is then padded to the FFT size set by the LDR\_CONF.SIZE bitfield. The LDR\_CONF2.PADF bitfield is used to determine how many zero data points are added to the front of the dataset. The remainder are added at the end.

Note that the dataset size cannot exceed the size set in the LDR\_CONF.SIZE bitfield. If the programmed value of LDR\_CONF2.PADF would violate this limit, then the behaviour of the loader is undefined but correct results will not be obtained. Similarly, LDR\_CONF.SIZE must not be set to a smaller number than the number of data points left after the Truncate operation.

When the FFT Accelerator is disabled by setting LDR\_CONF.FFTBYPS padding will still be performed unless LDR\_CONF2.PADDIS is also set. This allows data sets with non-power-of-two sizes to be passed to MATH2.

### 20.3.2.5 FFT Accelerator

The FFT accelerator uses a pipelined architecture to implement the efficient radix  $2^2$  architecture. It performs internal operations on 24-bit data. The 32-bit input data is reduced to 24-bit using the top bits (31:7) and rounding to 24 bits using the round to nearest, ties to even method. The output of the FFT accelerator is extended to 32 bits by appending 8 zero bits as LSBs.

The FFT accelerator can be completely bypassed using the LDR\_CONF.FFTBYPS bit.

When performing an FFT, the accelerator will divide the output by the number of data points in the FFT to ensure that the output data does not overflow the integer output format.

An inverse FFT can be performed by setting the LDR\_CONF.IFFT bit. Setting the mode to IFFT will disable the descaling function of the FFT accelerator IP and the output will not be divided.

**Note:** *The inverse FFT function is intended to be used to post-process FFT results to recreate the original input data. If arbitrary input data is used or windowing or scaling is applied to the FFT results before attempting an inverse FFT, an overflow may occur at the output of the FFT accelerator which would normally be compensated for by the descaling operation. This overflow will be signalled to the processor by interrupt if the STAT.INTMSK[1] bit is set.*

Note that the dataset size cannot exceed the size set in the LDR\_CONF.SIZE bitfield. The number of datapoints read from the Buffer RAM is determined by the Input DMA configuration. If the programmed values of LDR\_CONF.DRPF, LDR\_CONF.DRPL and LDR\_CONF2.PADF results in a number of datapoints that violate this limit, then the behaviour of the loader is undefined but correct results will not be obtained.

## Signal Processing Unit 2 (SPU2)

**Table 800 FFT Engine Configuration Parameters**

Parameter	Definition	Comments
FFT length	defined in the global registers as the number of operands for the inner loop	<b>LDR_CONF.SIZE</b>
FFT enable	FFT enabling / disabling	logically the same as “not FFT bypass” <b>LDR_CONF.FFTBYP</b>
Inverse FFT Mode	Configure the FFT accelerator to perform and Inverse FFT operation instead of the normal FFT	<b>LDR_CONF.IFFT</b>

### 20.3.3 Data Unloader

The Data Unloader writes the data from the FFT accelerator module into the output buffer RAM. All user programmable options are associated with the sideband processing of signal power to generate a histogram. By default, the Data Unloader writes 32 bit precision complex data to the Buffer RAM. As a space saving measure, the UNLDR\_CONF.FORMAT bitfield can be set to 16BIT and the UNLDR\_CONF.EXPNT bitfield used to set a common exponent value for the written data. The 32 bit values will then be shifted right by the number of bit positions programmed in the UNLDR\_CONF.EXPNT field before a saturating truncation to the 16 LSBs is performed (see “[Saturating Truncation” on Page 41](#) for further details). When using this mode, the value of the EXPNT should be restricted to values between 0 and 16. The behaviour of the Data Unloader with values outside this range is undefined.

If writing of 32 bit precision data is configured, then setting the EXPNT field to a non-zero value will cause the FFT output data to be shifted left by the number of bits set in the EXPNT field (preserving sign). If this causes the data value to overflow, then the data will be set to full scale minimum or maximum value depending on the sign.

*Note:* *If the data being processed is 32 bit precision linear power (ID\_RM\_CONF FORMAT=PWR32BIT or PWR16FP, then compression is not supported and the data format (UNLDR\_CONF FORMAT) should always be set to 32BIT.*

The Data Unloader also tracks the number of sign bits (i.e. leading ones and zeros) in the FFT output data for use by the application software.

**Table 801 Unloader Configuration Parameters**

Parameter	Definition	Comments
Common Exponent	The 32 bit data will be shifted right by this number of bits before truncation to the 16 LSBs if the FORMAT selected is 16 bit and left if the format is 32 bit	<b>UNLDR_CONF.EXPNT</b>
Format	Select the format of the data to be written to the buffer memory. Options are 16 or 32 bit precision complex.	<b>UNLDR_CONF FORMAT</b>

---

## Signal Processing Unit 2 (SPU2)

### 20.3.3.1 Power Histogram

The power histogram module stores a histogram of the FFT output power distribution into the configuration RAM. The method used derives the histogram bin index from the  $\log_2$  of the power. This method first requires the linear power to be calculated.

The linear power value used is derived in two possible ways:

- If the data format is complex, it will be calculated to 32 bit precision using the method described in “[Pre-Processing Units](#)” on Page 40. This method ensures that all 32 bits of the calculated power value contain useful data (i.e. all bits will toggle, no bits will be at a constant value over the full range of inputs)
- If the data format is 32 bit power, the data will be used directly.

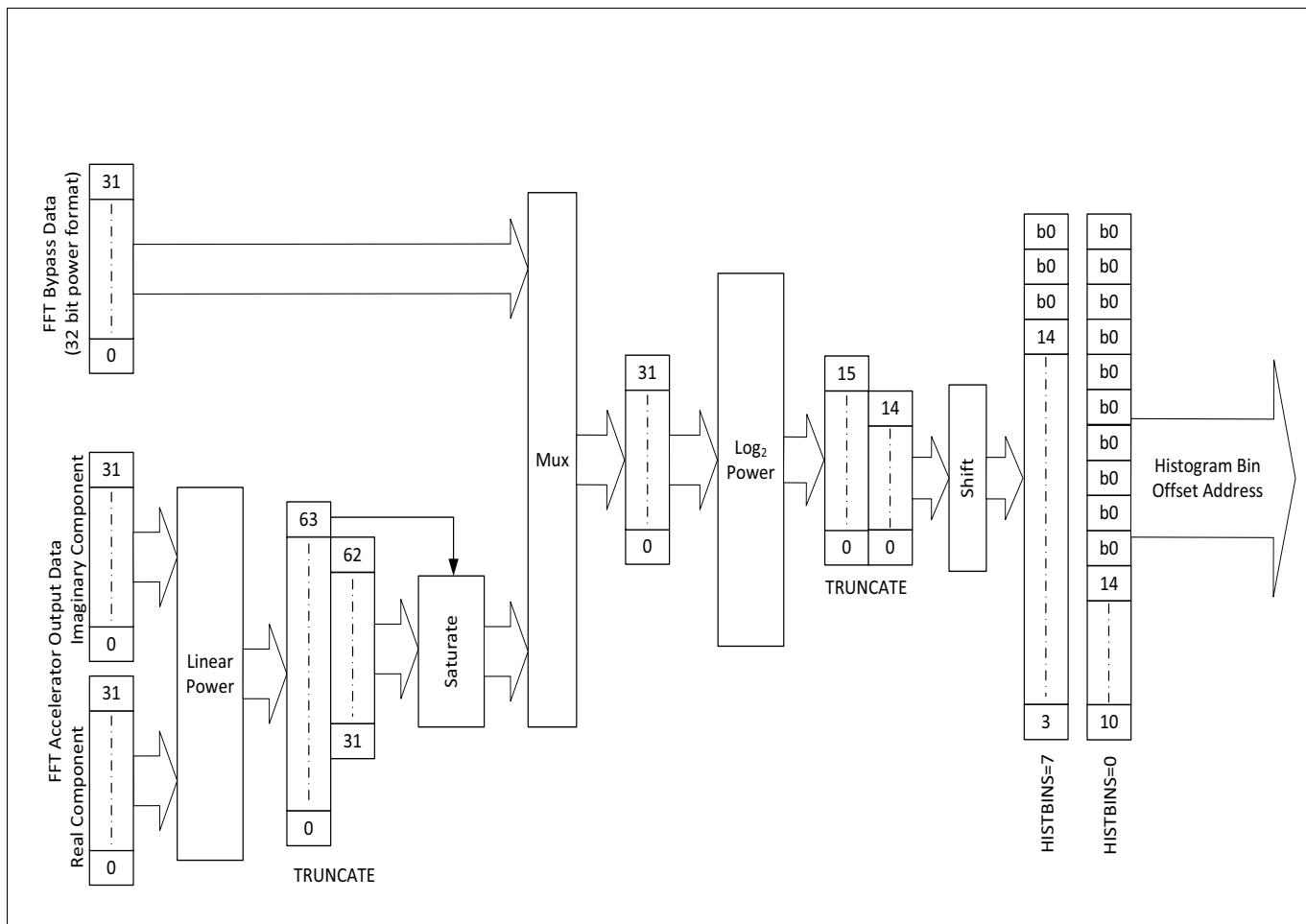
**Note:** *The value of the ODP\_CONF.FTR bitfield is considered when calculating the power values used for the histogram. If ODP\_CONF.FTR is set, then the imaginary component of the FFT output data will be set to  $0_D$ .*

This 32 bit linear power value is used to derive a 16 bit  $\log_2$  power value. The  $\log_2$  power value is formatted as:

- bit 15: sign bit, always 0 (inputs of less than  $2^0$  are mapped to a result of 0)
- bits 14 downto 10: integer part
- bits 9 downto 0: fractional part

The  $\log_2$  power value is then right shifted by a programmable number of bits and used as a word address offset from the histogram base address in the configuration memory. The 64 bit word stored at this address is incremented. As bit 15 is always zero, it is specifically excluded from the address

## Signal Processing Unit 2 (SPU2)



**Figure 260 Diagrammatic Representation of Histogram Bin Offset Generation**

The histogram is enabled by writing an  $1_B$  to UNLDR\_CONF.HIST\_EN, the base address is stored in UNLDR\_CONF.HISTBASE. The number of bins is configured by the value in UNLDR\_CONF.HISTBINS. A minimum of 32 bins and a maximum of 4096 bins are supported. The number of bins must always be a power of 2 and is given by  $2^{(\text{HISTBINS}+5)}$ . The size of the right shift referred to in the previous paragraph is therefore (10-HISTBINS).

The byte address of any given bin is UNLDR\_CONF.HISTBASE + ((BIN NR.) \* 8).

The power histogram will be calculated for the number of FFTs controlled by the values in the UNLDR\_CONF2.START and UNLDR\_CONF2.END fields. The FFTs of the measurement cycle are counted with the first FFT being FFT 0. If the count is greater than or equal to UNLDR\_CONF2.START and less than or equal to UNLDR\_CONF2.END, then the FFT is used to compute the histogram. If UNLDR\_CONF2.END is set to all 1, then accumulation of histogram data will run continuously from when the count reaches the UNLDR\_CONF2.START value until the end of the measurement cycle.

For every FFT included in the histogram, only those bins in the range defined by UNLDR\_CONF3.STARTBIN and UNLDR\_CONF3.ENDBIN are included in the calculation. The range is inclusive with the first bin being number 0 and setting UNLDR\_CONF3.ENDBIN to any value greater than the last bin number has the same effect as setting it to include the last bin.

The calculation of the power histogram can be further filtered by considering antenna ID. This mode is enabled by setting UNLDR\_CONF.HAFE. If set, only FFT results associated with the antenna ID stored in the UNLDR\_CONF.AVF will contribute to the power histogram or update the FFT count used for checks against UNLDR\_CONF2.START and UNLDR\_CONF2.END values.

## Signal Processing Unit 2 (SPU2)

The histogram needs two clock cycles to process each bin as it needs to read the current value from the configuration memory and then write it back after updating. Consequently, the histogram must only be enabled if the FFT clock rate is less than one half, that is AUXCTRL.DIVF is set to something greater than or equal to  $8_D$ .

**Note:** *The histogram and the windowing function in the LOADER both require full bandwidth access to the Configuration RAM while the TC3Ax SPU is running. To prevent conflict, the Configuration Memory is split into four towers. See “[Configuration Memory Usage Restrictions](#)” on Page 68 for details. The memory used for histogram and the memory used for the window functions must not be in the same tower if both functions are active at the same time. The memory allocated for the histogram must not be split between towers under any circumstances as this is not supported by the hardware implementation.*

**Table 802 Histogram Configuration Parameters**

Parameter	Definition	Comments
Histogram enable	enable / disable for histogram	<b>UNLDR_CONF.HISTEN</b>
Number of classes	from 32 to 4096 classes	<b>UNLDR_CONF.HISTBINS</b>
Base address of the histogram in the RAM		note that we may have a modulo here to avoid too high complexity in address computations <b>UNLDR_CONF.HISTBASE</b>
Start Delay	FFTs to count before starting histogram calculation	<b>UNLDR_CONF2.STRT</b>
End Delay	FFTs to count before ending histogram calculation	<b>UNLDR_CONF2.END</b>
Start Bin	Number of the first bin to include	<b>UNLDR_CONF3.STRTBIN</b>
End Bin	Number of last bin to include	<b>UNLDR_CONF3.ENDBIN</b>
Filter by Antennae	Enable histogram data to be accumulated only for a single antenna	<b>UNLDR_CONF.HAFE</b>
Antenna Number to use for filtering	Index of Antenna to be used for histogram data accumulation	<b>UNLDR_CONF.AVF</b>
Real Data Format	Ignore imaginary component when computing power	<b>ODP_CONF.FTR</b>

### 20.3.3.2 Statistical Information

The Data Unloader can calculate the mean value of the power for each FFT while processing the data. The information will be stored in a FIFO for later use by the Output Data Processor. Calculation and storage of this information is enabled using the SBCTRL.EN bitfield.

The mean is always calculated from 32 bit data. If 16 bit complex format is selected by UNLDR\_CONF.FORMAT, the mean calculation will use the 32 bit data before the shift defined by UNLDR\_CONF.EXPNT is applied. If 32 bit complex format is selected, the mean calculation will use the 32 bit data after the shift defined in UNLDR\_CONF.EXPNT is applied. This ensures that the mean is calculated using the same data as used by the Output Data Processor.

The linear power value used is calculated using the method described in “[Pre-Processing Units](#)” on Page 40.

## Signal Processing Unit 2 (SPU2)

This gives half the information needed to compute the variance (square of standard deviation). To complete the variance calculation, a second pass through the data is needed. This is done when the data is read by the Output Data Processor.

Statistical information for the measurement cycle can be stored in the Radar Memory. One set each of minimum power, maximum power, mean power and standard variance will be stored for each FFT processed.

The division used to complete the variance<sup>1)</sup> calculation is a “shift right” operation rather than a full hardware division. This causes some inaccuracies when the number of samples is not a power of two. This does not affect the statistical information computed from FFT results but the calculation of the statistical information uses a mean value based on the number of samples being rounded up to the next power of two when the FFT engine is bypassed ( $LDR\_CONF.FFTBYP = 1_B$ ) and padding is disabled ( $LDR\_CONF.PDDIS = 1_B$ ).

These operations are described in more detail in the relevant section in the Output Data Processor section.

**Note:** *To ensure that the data used for the statistical calculations is consistent the value of the ODP\_CONF.FTR bitfield is considered when calculating the power values used for the computing the mean. If ODP\_CONF.FTR is set, then the imaginary component of the FFT output data will be set to 0\_D.*

### 20.3.4 Streaming Processor 2, The Output Data Processor

The Output Data Processor module formats and writes the results datasets to the Radar Memory. It can also route the data through multiple processing modules working on complex data and power. The Output Data Processor is split into two submodules, the MATH2 Unit and the Output DMA Engine. When reading data from the Buffer RAM, the Processor uses the UNLDR\_CONF.FORMAT and UNLDR\_CONF.EXPNT to determine whether the stored data is in native 32 bit precision format or has been compressed to 16 bit precision and requires reformatting.

The ODP has a processing path for manipulating the base data and bin rejection working in parallel with multiple analysis units (CFAR, integration (NCI/DBF), statistics and arithmetic).

#### 20.3.4.1 Streaming Processor 2 Data Fetch from Buffer Memory

There are two classes of processing path through the MATH2 which require the data to be read from Buffer RAM in two incompatible ways.

- Operations such as the FFT data output path through the bin rejection unit require the data to be read as consecutive points from each FFT result.
- Operations using the data integrated across antenna requires equivalent data points to be read consecutively from each FFT result stored in the buffer memory so the integrated value can be calculated.

If both types of path are enabled by the programmed configuration options, then this conflict is addressed by reading the data from the buffer memory twice. The first fetch will be used for the operations requiring linear addressing by FFT result. The second fetch will be performed for operations requiring the integrated result across antennae. The number of fetches required is evaluated and executed independently for each pass if double pass mode is enabled.

#### 20.3.4.2 In Place FFT

Setting ODP\_CONF.IPF configures the Output Data Processor to generate addresses using the same base and offset scheme used by the input DMA. To support this registers are available to allow the offsets and loop repeat counts to be configured. The intention is that the output data can be written to the memory locations cleared by the reading of the input data.

1) The variance calculated is the population variance rather than a sample variance. The sum of the squares is divided by the number of samples rather than the number of samples minus one.

## Signal Processing Unit 2 (SPU2)

To do this the Output DMA uses an equivalent set of counter and offset registers to those instantiated in the Input DMA engine. The parameters used to configure this duplicate register set are derived from the Input DMA configuration registers and the values should be calculated so that the Output DMA addresses follow the same sequence as the Input DMA addresses.

There are some differences in the way the ODM register values are used to generate addresses:

- All offset fields are EMEM word addresses
- There are no equivalents to the IDM\_RM\_CONF.TRNSPS and ID\_RM\_CONF.PM bitfields. Dependent on the settings of these bitfields, one of the loop repeat registers will be incremented by more than one. As no equivalent for these bitfields exist in the equivalent ODM registers, the equivalent loop repeat value will have to be divided to compensate. The increment value is the number of samples in a 256 bit EMEM word, the equivalent ODM value will be the IDM bitfield value divided by the number of samples in a 256 bit word.

The Output DMA will always assemble enough results to completely fill each 256 bit word before moving to the next address in the sequence. This will result in fewer words being written than read if the output data size is smaller than the input data size. Conversely, if the output data requires more memory than the input data, results will be unpredictable and, consequently, use of in place FFT in these cases is not supported.

In Place FFT addressing only applies to the writing of the FFT results.

It is strongly recommended that bin rejection is not used with in place FFT when using transposed addressing as no additional memory saving will be possible (data will be written as a sparse array in memory) and addressing of the resultant data will be extremely complex.

### 20.3.4.2.1 Restrictions

Correct operation of this mode requires that the Input DMA reads and uses all 32 bytes of each word in Radar Memory in a single pass. If only part of the word is used, it is probable that the Output Data Processor will overwrite the location before a further pass through the data cube by the Input DMA uses the rest of the word.

This mode cannot be used if Double Pass mode is used to write two sets of FFT results.

### 20.3.5 MATH2 Unit

The MATH2 Unit implements multiple parallel processing units as shown in [Figure 261](#) below. These processing units are grouped into two data flows, one working on the raw complex data and the other on linear or  $\log_2$  power. The following table shows the operating domain and concurrency of the various functions:

**Table 803 MATH2 Unit Functional Concurrency**

	Function - Only one function can be active for each O/P Channel								
	Stats	Local Max/ Logical Func	CFAR CA	CFAR GOS	FFT Data <sup>1)</sup>	Log <sub>2</sub> Signal Power	ND0 <sup>2)</sup>	ND1 <sup>2)</sup>	Label list
Power Domain <sup>3)</sup>	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
Complex Domain <sup>4)</sup>	No	No	No	No	Yes	No	Yes	Yes	No
O/P Channel	PORT3	PORT8	PORT4	PORT5	PORT1	PORT2	PORT6	PORT7	PORT9

1) Incorporates any or all of scalar add, scalar multiply and bin rejection.

2) ND0 and ND1 are the dual integration units that operate in power/complex domain depending on their configuration

3) Function is possible using power as input

## Signal Processing Unit 2 (SPU2)

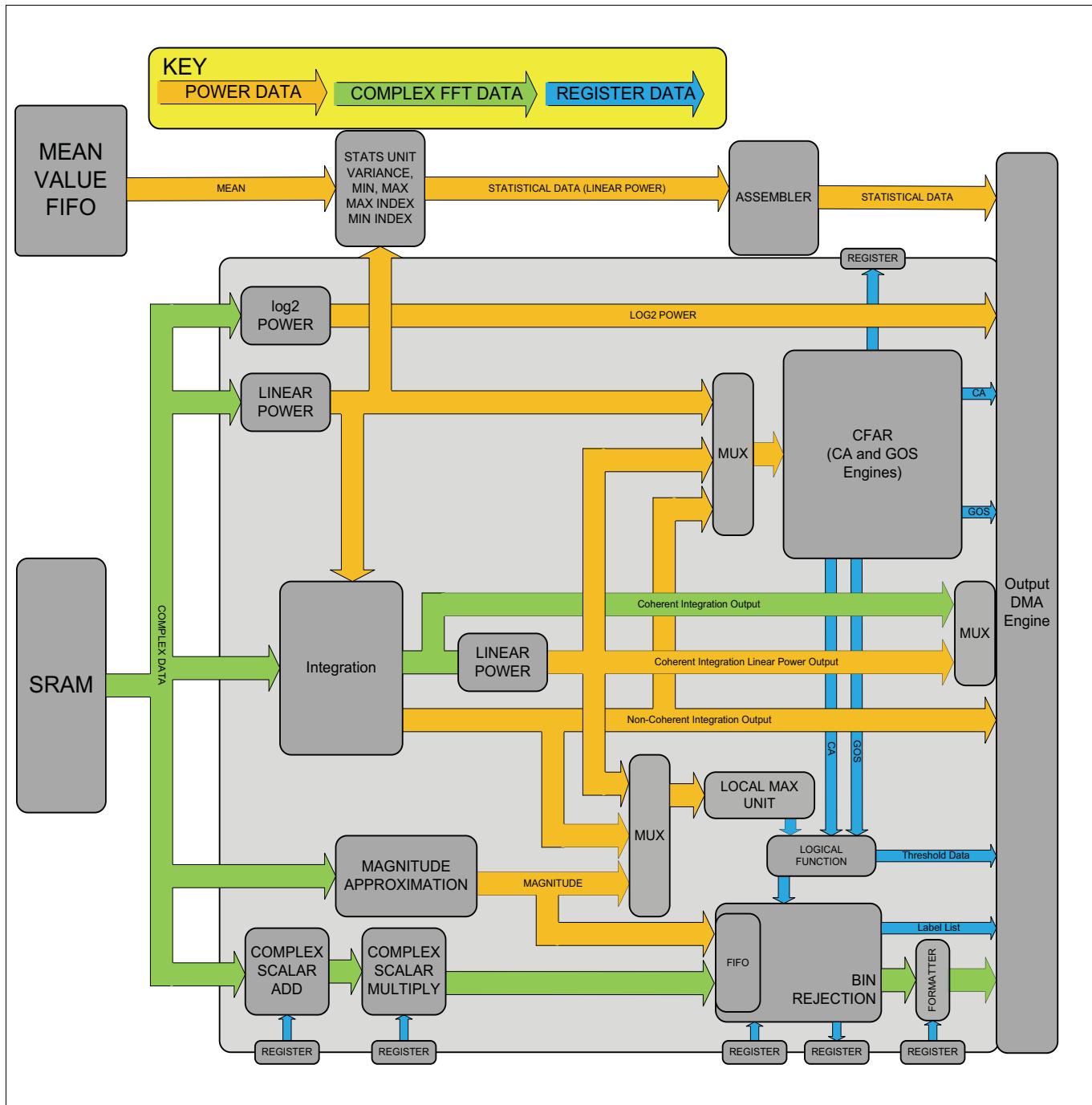
- 4) Function is possible with Complex Values as input

**Table 804 Output DMA Engine Data Streams**

Output port	Definition	Comments	Precision (Bits)	Padding <sup>1)</sup> per FFT or measurement cycle
Port1	3 sources of informationFFT unitFFT BIN rejection and zeroingadd scalar or complex to vector		16 or 32 bit precision, complex or real formats	FFT
Port2	$\log_2$ signal power		16 bits	FFT
Port3	min, max, average, (standard deviation) <sup>2)</sup>		32 bits	FFT
	index of min, index of max	12 bits of data, padded to 16 bits with leading zeroes	16 bit	
Port4	CFAR (CA Engine)		1 bit per FFT bin	cycle
Port5	CFAR (GOS Engine)		1 bit per FFT bin	cycle
Port6	Vector weighted sum, complex or power domain	typically: coherent or non coherent integration <sup>2)</sup>	16 or 32 bit complex 16 or 32 bit power	FFT
Port7	Vector weighted sum, complex or power domain	typically: coherent or non coherent integration <sup>2)</sup>	16 or 32 bit complex 16 or 32 bit power	FFT
Port8	local max or logical function		1 bit per FFT bin	cycle
Port9	Label list output	Amount of data depends on CFARCFG3.	64 bits max per label	cycle

- 1) Padding is used to ensure that the next set of data written starts on a 256 bit aligned address. Data intended to be read back into the TC3Ax SPU will have one element per FFT bin processed and each dataset must start on a 256 bit aligned address. These data streams will be padded per FFT. Other data streams will be padded per measurement cycle to optimise memory usage
- 2) Output from this operation is a 1D vector of the same length as the input FFTs

## Signal Processing Unit 2 (SPU2)



**Figure 261 MATH2 Unit Architecture**

The MATH2 unit will always expand the data read from the buffer memory to 32 bit precision before using it in any functions. The expansion function is the inverse of the compression operation used in the Unloader with any additional LSBs being padded with  $0_B$  and the MSBs being sign extended. See [Section 20.3.3, “Data Unloader” on Page 32](#).

Where 32 bit precision values are multiplied to generate a result at 32 bit precision, rescaling of the result will take place after the multiplication operation. Rescaling will always remove the least significant bits of the result.

When multiplying two signed 32 bit numbers, the result will be a 64 bit signed value and this will be scaled to 32 bits by dividing by  $2^{31}$  and rounding down to the nearest integer while discarding the MSB. As multiplying two full scale negative numbers will result in an overflow using this algorithm, this corner case will generate a full scale, positive result.

## Signal Processing Unit 2 (SPU2)

When multiplying two unsigned 32 bit numbers, the result will be a 64 bit unsigned value and this will be scaled to 32 bits by dividing by  $2^{32}$  and rounding down to the nearest integer.

If any calculation result is scaled to 16 bits, then the result will be rounded to the nearest integer after division. Results which fall exactly midway between two integers will be rounded away from zero.

Scaling of the results of 32 bit addition operations to produce a 32 bit output are handled in a similar manner. The intermediate result is always sized so that overflow does not occur and then this is scaled down to 32 bits by dividing and rounding down towards -Infinity to the nearest integer. The scaling to 32 bits will, in most cases, occur automatically unless the number of operands is configurable, in which case, a configurable scaling factor has been provided.

### 20.3.5.1 Pre-Processing Units

The pre-processing units convert the raw FFT data to  $\log_2$  power, linear power or magnitude or change the precision of the data.

#### 20.3.5.1.1 Linear Power Calculation

Linear power is calculated to 32 bit precision by

- squaring the real and imaginary components and scaling by dividing by  $2^{31}$ .
  - For full scale positive data, this gives a value of  $7FFF\_FFFE_H$
  - For full scale negative data, this gives a value of  $8000\_0000_H$
- adding the truncated squares of the real and imaginary components and saturating the results
  - $7FFF\_FFFE_H$  for both real and imaginary complex components gives  $FFFF\_FFFC_H$  when the squares are added and saturation does not take place
  - $8000\_0000_H$  for both real and imaginary complex components gives  $1\_0000\_0000_H$  when the squares are added and  $FFFF\_FFFF_H$  after saturation.

#### 20.3.5.1.2 $\log_2$ Power Calculation

$\log_2$  power is computed to a precision of 16 bits and will be stored as an unsigned integer value with 7 integer bits and 9 fractional bits. This is a representation of the 32 bit precision linear power value scaled to 64 bits by multiplying by  $2^{31}$ . This is done to be compatible with the  $\log_2$  power calculation used inside the CFAR module which uses a 64 bit linear power input. An input linear power value of  $2^{31}$  scales to a 64 bit value of  $2^{62}$ . A 64 bit value of  $2^{62}$  corresponds to a  $\log_2$  value of  $7C00_H$ . The MSB of the 16 bit value is always zero.

The algorithm used is as follows:

- In the first stage, the location of the first bit set in the 64 bit input value is used to determine the integer component of the output.
- In the second phase of the algorithm, the most significant eight bits of the residual from the first phase is used as the index into a 256 entry look up table used to apply an appropriate fractional correction to the integer value

The maximum granularity of the look up table output is 0.006. An input value of  $0000\_0000_H$  is mapped to an output value of  $0000_H$ .

## Signal Processing Unit 2 (SPU2)

### 20.3.5.1.3 Magnitude Approximation

For magnitude, the approximation:

(20.4)

$$\text{MAGNITUDE} = \alpha \times (\text{MAX}(\text{ABS}(I), \text{ABS}(Q))) + \beta \times \text{MIN}(\text{ABS}(I), \text{ABS}(Q))$$

is used. The two multiplication constants are stored in the ALPHA and BETA bitfields of the MAGAPPROX register. The magnitude value is calculated at 32 bit precision. ALPHA and BETA are unsigned constants scaled to have a value range of between 0 and 1 approximately (actually 0.99998 to five significant digits). The output is a 32 bit value scaled to have an overall gain of 1.

**Note:** *The ABS function used in the magnitude approximation maps a full scale negative number  $8000\_0000_H$  to a magnitude of  $7FFF\_FFFF_H$*

Later operations then use the magnitude value. Operations which use magnitude data are:

- Threshold Detection, [Section 20.3.5.3.3](#)
- Local Maximum, [Section 20.3.5.2](#)

### 20.3.5.1.4 Saturating Truncation

The saturating truncation operation will remove the 16 MSBs of a 32 bit quantity to leave a 16 bit result. The intention is that the 32 bit quantity is scaled such that no significant information is lost when doing this (i.e. the 17 MSBs contain only sign information). However if this not the case and one or more of the MSBs contain  $1_B$  for positive quantities or  $0_B$  for negative quantities, then the 16 bit result will be set to full scale positive in the first case and full scale negative in the second case.

### 20.3.5.2 Local Maximum Detection Unit

The module checks to see if the bin being checked is a local maximum by comparing against adjacent bins. If it is, then the bin will be flagged. The module can also simultaneously check the bin against a threshold value. The output information can be written to Radar Memory and used by the Bin Rejection Unit.

This module can be used concurrently with the CFAR engines. The Local Maximum Detection Unit has the same options for range and velocity spectrum extension as the CFAR engines. See [Section 20.3.5.5.8, CFAR spectrum extension](#) for a full description. It has the same control for its data source as the CFAR engines, see [Section 20.3.5.5.4, CFAR Module Configuration](#), except that the FFT input for Local Maximum comes from the Magnitude Approximation unit.

The Output DMA channel for output to memory is shared between Local Maximum and the [Logical Function](#) unit (see [Section 20.3.5.3.3](#)) so writing Local Maximum to memory requires CFARCTRL.LFOSEL to be set to LCLMAX.

**Table 805 Local Maximum Configuration Parameters**

Parameter	Definition	Comments
local max enable	enable / disable	<b>CFARCTRL.LCLMAX</b>
Radar Memory Address	Base address to be used for writing the threshold results to the Radar Memory	<b>LCLMAXBASE.BASE</b>
Output enable	Write data to Radar Memory	<b>LCLMAXBASE.EN</b>
Output select	Select whether LCLMAX is output or Logical Function	<b>CFARCTRL.LFOSEL</b>

## Signal Processing Unit 2 (SPU2)

**Table 805 Local Maximum Configuration Parameters (cont'd)**

Parameter	Definition	Comments
Mode	Select whether threshold, local maximum or both checks shall be active	<b>LCLMAX.MODE</b>
Local Max Check Combining	Logical Operation used to combine check results when both checks are active	<b>LCLMAX.CMBN</b>
Spectrum Extension Mode	Off, Range or Velocity	<b>CFARCTRL.EXTNSN</b>
Window Width <sup>1)</sup>	Window size to be used for local max calculation	<b>LCLMAX.WIDTH</b>
Threshold Value	user defined value	<b>LCLMAX.TSHLD</b>
Threshold Extension Mode	Left or Right Justification of the TSHLD value in the 32 bit comparison value	<b>LCLMAX.LJUST</b>
Input selection	select input from FFT engine, INT0 or INT1. Selecting INT0 or INT1 when that unit is not active is not supported and will produce undefined results.	note that when input from FFT engine is used, the approximate magnitude will be computed on each received BIN. <b>CFARCTRL.SRCSEL</b>

1) WIDTH must be set to  $1_B$  or  $2_B$  to flag any bins when MODE only has Local Max enabled

The Unit can be configured to flag bins where the magnitude is local max or where the magnitude is not a local max. The unit can simultaneously be configured to check against a threshold value and flag a bin if the magnitude is above the threshold or if the magnitude is below the threshold.

If both checks are enabled, the results are combined as defined using the LCLMAX.CMBN bit. Both logical AND and logical OR operations can be selected: for example flag a bin on both checks triggering a reject or flag a bin on either check triggering a reject.

The comparison is done with 32 bit values and the 24 bit LCLMAX.TSHLD is extended to 32 bits before being used. This extension can either be right or left justified. If the LCLMAX.LJUST bitfield is set to  $0_B$ , the extension is right justified and 8 MSBs of value  $0_B$  are appended. If LCLMAX.LJUST is set to  $1_B$ , the extension is left justified and 8 LSBs of value  $0_B$  are appended.

**Note:** The Spectrum Extension function ([Section 20.3.5.5.8](#)) is used to provide the extra window data needed for the Local Max Unit window if CFARCTRL.LCLMAX is set and LCLMAX.MODE is not equal to  $10_B$  or  $11_B$  meaning the local max function is active. CFARCTRL.SEWIN must be set to a value at least the LCLMAX.WIDTH value. If LCLMAX.WIDTH= $01_B$ , then SEWIN= $=000001_B$  and if LCLMX.WIDTH= $10_B$ , then SEWIN= $=000010_B$ .

### 20.3.5.3 FFT Data Output Path

This is the output path for the FFT results.

The base address for the FFT results data written to Radar Memory is stored in the ODP\_BASE register. The format of the data is controlled by the ODP\_CONF.MODE, ODP\_CONF.FTR, ODP\_CONF.SCALE, ODP\_CONF.EXPNT, ODP\_CONF.HPFP and ODP\_CONF.COMPRESS bitfields.

## Signal Processing Unit 2 (SPU2)

**Table 806 Output Data Processor Format Options**

Parameter	Definition	Comments
Enable FFT results output	On/Off switch to enable writing of the post-processed FFT results to memory	<b>ODP_CONF.MODE</b>
Results Compression	Enable Compression of the FFT results to 16 bit precision with a common exponent	<b>ODP_CONF.SCALE</b>
Results compression scaling factor	The scaling factor to be used if results compression is enabled. The 32 bit data will be shifted right by this number of bits before truncation to the 16 LSBs	<b>ODP_CONF.EXPNT</b>
Real Results	Force the complex component of the FFT results to 0 as it is read from the buffer memory. The complex component is written to memory but has a value of 0.	<b>ODP_CONF.FTR</b>
In Place Writing of Results	Configure the Output Data Processor to Compute Write addresses in the same order as the Input DMA so that results are written to the addresses cleared by reading input data	<b>ODP_CONF.IPF</b>
Write results as real only data	The FFT results written to memory do not have an imaginary component but consist only of packed, real values	<b>ODP_CONF.ROF</b>
Post-processing option for the FFT results	The FFT results can be converted to half precision floating point format (as defined in IEEE 754) before being written to memory.	<b>ODP_CONF.HPFP</b>
Compression of half precision floating point data	The FFT results in HPFP format can be compressed by packing 16 complex values into 256 bits.	<b>ODP_CONF.COMPRESS</b>

Operation of the HPFP bitfield is described in [Section 20.3.5.3.4, “Half Precision Floating Point Format” on Page 48](#). The compression is described in [Section 20.3.5.3.5, “FFT Data Compression” on Page 48](#).

### 20.3.5.3.1 Scalar Addition

The pre-processing unit can add a offset to each complex data value. The offset is added to each component, real and imaginary, of the data value. The value of the offset is a constant stored in the SCALARADD.OPERAND bitfield and the function will be activated if a non-zero value is written to this field.

### 20.3.5.3.2 Complex Rescaling

The pre-processing unit can multiply each data value by a scaling factor. This value is a 32 bit real constant stored in the SCALARMULT.OPERAND bitfield which allows the value to be rescaled but does not affect the phase. The function will be activated when a non-zero value is written to this field.

### 20.3.5.3.3 Bin Rejection Unit

This unit post-processes the data to be written to Radar Memory using the contents of the bin rejection register and the outputs from the CFAR engines and the Local Maximum Unit. The outputs of these units are combined in a logical function before being used for bin rejection.

The bin rejection register contains one bit for each “bin” (datapoint) in the FFT results. It is accessed through 64, 32 bit registers in the register space of the TC3Ax SPU. If the bit is set to 0, then the relevant bin will either be removed from the output dataset written to the Radar Memory or set to a value of  $0_D$ .

## Signal Processing Unit 2 (SPU2)

This feature can be used in conjunction with the logical function of the outputs of the CFAR engines and the Local Maximum Unit output<sup>1)</sup>. In this mode, the bin rejection register and the output from the Logical Function must both permit the bin to be written. If either function requires the bin to be removed (or set to zero), then it will be removed (or set to zero). For example, if the bin acceptance mask requires the bin to be set to  $0_D$ , then it will be set to  $0_D$  even if the Logical Function has identified it as a bin to be kept.

This can be viewed as the logical ANDing of a bin acceptance mask from the Logical Function with the acceptance mask programmed in the Bin Rejection register. Bins corresponding to bits set to  $1_B$  in the resulting, combined mask will be retained unchanged while bins corresponding to bits set to  $0_B$  in the resulting mask will be either removed from the output or set to a value of  $0_D$  depending on the mode of the bin rejection unit.

**Table 807 Bin Rejection Unit Configuration Options**

Signal	Definition	Comments
operating mode	$00_B$ = Off $01_B$ = BIN rejection, remove bins from output $10_B$ = Set BIN value to 0 $11_B$ = Reserved	<b>BINREJCTRL.RMODE</b>
BIN list to be removed	2048 bit register that defines which BIN is to be removed or zeroed (1 means BIN to be retained or passed unchanged)	<b>BINx.REJ.B</b> <b>x=0-63</b>

### Logical Function

The logical function unit combines the 3 logical outputs of the CA CFAR, GOS CFAR and Local Max<sup>2)</sup> units into a single bit that controls bin rejection along with the bin rejection register. The function uses the three input bits, where  $1_B$  means keep bin and  $0_B$  means reject bin, as an address composed of Local Max as the LSB, GOS as the middle bit and CA as the MSB. This address is used to select a bit from the binary LFUNC register field value. Some examples of how the Logical Function may be configured are given in the table below:

**Table 808 Configuring the LFUNC Parameter**

Operation	L FUNC Value	Comments
Always zero	$00000000_B$	Any input value selects $0_B$
<b>GOS OR CA OR LMAX<sup>1)</sup></b>	$11111110_B$	Output is $0_B$ only when all three inputs are $0_B$
<b>GOS AND CA AND LMAX</b>	$10000000_B$	Output is $1_B$ only when all three inputs are $1_B$
<b>GOS</b>	$11001100_B$	Output only depends on the value of GOS
<b>CA</b>	$11110000_B$	CA is the MSB of the address
<b>CA AND (GOS OR LMAX)</b>	$11100000_B$	As above but when both GOS and LMAX are $0_B$ the output is $0_B$
<b>GOS XOR LMAX</b>	$01100110_B$	The output is independent of CA so the bit pattern is duplicated in the LS 4 bits and MS 4 bits. When GOS and LMAX are both $1_B$ the output is $0_B$
Always one	$11111111_B$	Any input value selects $1_B$

1) LMAX represents the Local Maximum output

1) If INT0 or INT1 are selected as the source for CFAR and Local Maximum then Bin Rejection will ignore the Logical Function value in this case.  
2) As Local Max flags bins to be rejected, while CFAR flags bins to be kept, the Local Maximum logical value is inverted before input to the Logical Function.

## Signal Processing Unit 2 (SPU2)

Note: If a unit, CA CFAR, GOS CFAR or LCLMAX, is not enabled LFUNC should be configured such that its value has no effect on the Logical Function output. Failure to do this will result in undefined behaviour.

Note: If none of the units CA CFAR, GOS CFAR nor LCLMAX are enabled LFUNC should be set to  $1111111_B$ . Configuring to  $0000000_B$  will result in all bins being rejected.

The Output DMA channel for output of the bit stream to memory is shared between Logical Function and the **Local Maximum Detection Unit** (see [Section 20.3.5.2](#)) so writing the Logical Function output to memory requires CFARCTRL.LFOSEL to be set to LFUNC. The Logical Function output also feeds into the Label List output function (see below).

**Table 809 Logical Function Configuration Parameter**

Parameter	Definition	Comments
Logical Function	An 8 bit field representing a 3 bit input 1 bit output lookup table. The inputs are Local Max, CA CFAR and GOS CFAR. Setting the field to all ones makes the output of the Bin Rejection unit independent of any of the CFAR or Local Max units.	Function acts on the output of the CFAR and Local Max units <b>CFARCTRL.LFUNC</b>
Radar Memory Address	Base address to be used for writing the results to the Radar Memory. This is the address where the start of bit stream is.	<b>LCLMAXBASE.BASE</b>
Output enable	Enable output of data to Radar Memory. Note, if LFOSEL==LCLMAX and CFARCTRL.LCLMAX==OFF no data shall be written to memory. Likewise, if LFOSEL==LFUNC and none of CA CFAR, GOS CFAR nor LCLMAX are enabled.	<b>LCLMAXBASE.EN</b>
Output select	Select whether LCLMAX is output or Logical Function	<b>CFARCTRL.LFOSEL</b>

### Label List Output

The Label List function operates on Label Events. A Label Event is defined based on the task being executed by MATH2. For an integration task (ND\_CTRL.MODE is not OFF and CFARCTRL.SRCSEL==INT0 or INT1) a Label Event is defined as the output of the Logical Function being a one. For a linear fetch task (CFARCTRL.SRCSEL==FFT) a Label Event is defined as the combination of the Logical Function being one and the Bin-Rejection mask set to accept a bin. As these two cases are mutually exclusive, given CFARCTRL.SRCSEL can either be FFT or not FFT, the Label List function will only operate in one of the tasks if two fetch modes are needed to fulfil the requirements of a configuration.

A record of Label Events may be written to Radar Memory as a list of labels by enabling Label List output. Each label has two possible parts: the system address and a 2D index. Either or both may be enabled by CFARTRLLBLCON. When both are enabled the system address is written first.

The system address and index (x,y) is calculated as shown in the pseudocode below.

```

x := 0
y := 0
address := LBLCALC.BASE<<5
FOREACH bin
    IF Label_Event

```

## Signal Processing Unit 2 (SPU2)

```

        write(address)
END
IF x < LBL.XRPT
    x := x + 1
    address := address + LBLX.MUL
ELSIF y < LBL.YRPT
    x := 0
    y := y + 1
    address := address + LBLY.MUL
ELSE
    x := 0
    y := 0
    address := address + LBLZ.MUL
END
END

```

The value of LBLCALC.BASE is shifted left 5 places to convert from a 256-bit word address to a system byte address. The first data point in the 3D data cube is usually located at this address. The address calculations are done in 32-bit arithmetic to produce a system byte address that can be written to Radar Memory.

The 2D index (x,y) is formatted as a 32 bit word with the x value in bits 12:0 and the y value in bits 25:13. Bits 31:26 of the word are always zero. The index output is only useful when CFARCTRL.SRCSEL==INT0 or CFARCTRL.SRCSEL==INT1. In this case the FOREACH iterates over the single results produced by integrating over multiple antenna so there are only two dimensions. If CFARCTRL.SRCSEL==FFT the 2D index (x,y) may not, depending on the settings of LBL.XRPT and LBL.YRPT, contain enough information to uniquely identify the bin of interest.

For each sample when there is a Label Event, this function appends one or two 32 bit words, depending on the value of CFARCTRL.LBLCON, to the data to be written to memory. The data is written starting at the location defined by the base address, LBLBASE.BASE. A count of labels written is maintained in two counters MD0\_LBLCNT and MD1\_LBLCNT to go with the two passes available in Double Pass Mode (see “[Double Pass Mode](#)” on Page 16).

The labelling function is enabled when CFARCTRL.LBLMODE is set to one of three active values:

- COUNT: just count Label Events
- NOINT: just write data
- INT: write data and generate interrupts in both of the following cases:
  - MDq\_LBLCNT.CNT reaches half the value stored in CFARCFG3.LBLOLM
  - MDq\_LBLCNT.CNT reaches the value stored in CFARCFG3.LBLOLM

The count is cleared by writing 1<sub>B</sub> to the MDq\_LBLCNT.CLR bitfield. The count should only be cleared when the SPU is idle.

The behaviour for modes other than COUNT after the count reaches CFARCFG3.LBLOLM is controlled by CFARCFG3LBLWRAP. If this bit is set the next Label Event after the count reaches CFARCFG3.LBLOLM will cause the following actions:

- any pending data is padded with zeros to width of the memory interface and written to Radar Memory at the current location
- MDq\_LBLCNT.CNT is reset to zero
- writing address is set to LBLBASE.BASE

before any more words are added to the data to be written.

When data output is enabled and CFARCFG3LBLWRAP is set, CFARCFG3.LBLOLM must be set such that:

## Signal Processing Unit 2 (SPU2)

$(\text{LBLLOLM} + 1) \bmod 8 = 0$

This is to ensure that the write address only wraps at a complete Radar Memory words boundary.

If CFARCFG3.LBLWRAP is not set no data shall be appended to the data to be written after the word that corresponds to the case: MDq\_LBLCNT.CNT==CFARCFG3.LBLLOLM.

At the end of processing a measurement cycle, in any case when data writing is enabled, any remaining data to be written to memory shall be padded with zeros to width of the memory interface and flushed to Radar Memory.

**Table 810 Label List Output Configuration Parameters**

Parameter	Definition	Comments
Label mode	Selects if the unit is enabled and what effect it has	Function labels CFAR and Local Max results <b>CFARCTRL.LBLMODE</b>
Label content select	Selects what is written as the label data. This may be the system address of the peak, the 3D index or both	Function labels CFAR and Local Max results <b>CFARCTRL.LBLCON</b>
Base address	Base address for data in Memory	<b>LBLBASE.BASE</b>
Label Calculation Base address	Value used as the base for label address calculation	<b>LBLCALC.BASE</b>
Label Calculation X multiplier	Value used as an address increment for the X dimension	<b>LBLX.MUL</b>
Label Calculation Y multiplier	Value used as an address increment for the Y dimension	<b>LBLY.MUL</b>
Label Calculation Z multiplier	Value used as an address increment for the Z dimension	<b>LBLZ.MUL</b>
Label Calculation X repeat	The number of elements in the X direction	<b>LBL.XRPT</b>
Label Calculation Y repeat	The number of elements in the Y direction	<b>LBL.YRPT</b>
Counter	Count of accepted bins	<b>MDq_LBLCNT.CNT</b>
Counter clear	Write a $1_B$ to clear the count	<b>MDq_LBLCNT.CLR</b>
Count limit	The maximum number of flagged bins to output as labels. The maximum number output is the value of this field plus one.	<b>CFARCFG3.LBLLOLM</b>
Wrap enable	Wrap the count to zero and address to LBLBASE.BASE	<b>CFARCFG3.LBLWRAP</b>

### Threshold Detection

The threshold detection function is integrated into the Bin Rejection Unit. The function checks to see if the bin being checked exceeds a programmed threshold. If the threshold is exceeded, then the bin is set to zero. The threshold comparison is based on the magnitude of the bin.

This is a 32 bit comparison. The 24 bit BINREJCTRL.VALUE is extended to 32 bits before being used. This extension can either be right or left justified. If the BINREJCTRL.LJUST bitfield is set to  $0_B$ , the extension is right justified and 8 MSBs of value  $0_B$  are added. If BINREJCTRL.LJUST is set to  $1_B$ , the extension is left justified and 8 LSBs of value  $0_B$  are added.

## Signal Processing Unit 2 (SPU2)

**Table 811 Thresholding Configuration Parameters**

Parameter	Definition	Comments
thresholding enable	enable / disable. If enabled, set a bin to zero if it exceeds the magnitude defined in BINREJCTRL.VALUE	Function is Combined into the Bin Rejection Unit <b>BINREJCTRL.ZMODE</b>
threshold value	user defined value	<b>BINREJCTRL.VALUE</b>
Threshold Extension Mode	Left or Right Justification of the VALUE value in the 32 bit comparison value	<b>BINREJCTRL.LJUST</b>

### 20.3.5.3.4 Half Precision Floating Point Format

The FFT results can be reformatted in IEEE 754 binary16 (half precision) floating point format. This is enabled by setting the bit **BEx\_ODP\_CONF (x=0-1).HPFP** to  $1_B$ . The outputs from Non-Coherent Integration and Digital Beam Forming also support this format.

The half precision floating point format has:

- a sign bit
- 5 bits of exponent encoding a value between -14 and 15
- 11 bits of significand precision<sup>1)</sup>

With this format, the magnitude which can be encoded ranges from a minimum of  $1 \times 2^{-14}$  (neglecting subnormal numbers<sup>2)</sup>) to a maximum of  $(2-2^{-10}) \times 2^{15}$  which is equivalent to  $2^{16}-2^5 = 65504$ .

The maximum range of a 32 bit signed integer is from  $-2147483648$  ( $0-2^{31}$ ) to  $2147483647$  ( $2^{31}-1$ ). To best fit this into the range of the half precision floating point format, the 32 bit precision number is first divided by  $2^{15}$  to give a range of  $-65536$  ( $0-2^{16}$ ) to (neglecting the fractional component<sup>3)</sup>  $65535$  ( $2^{16}-1$ ). This is then converted to the half precision floating point format with values less than -65504 or greater than 65504 saturating at the relevant limit.

For unsigned 32-bit power data the number is first divided by  $2^{16}$  and then treated the same as signed data.

The rounding used during these conversions is “round to nearest, ties to even”.

### 20.3.5.3.5 FFT Data Compression

The FFT data in half precision floating point format may be compressed so that 16 complex values fit into one 256 bit word in memory. This is enabled by setting the bit **BEx\_ODP\_CONF (x=0-1).COMPRESS** to  $1_B$ . The ODP\_CONF.ROF, ODP\_CONF.SCALE and ODP\_CONF.EXPNT bitfields have no effect if this mode is enabled. If this mode is enabled the ODP\_CONF.IPF field may only be set to  $1_B$  if RM\_CONF2.FORMAT==COMPRESS otherwise the behavior is undefined. The compressed format only applies to complex FFT data. The compression is lossy.

### 20.3.5.4 Dual Integration Units

The two Integration units are both capable of performing Non-Coherent Integration (NCI) or Digital Beam Forming (DBF), also known as Coherent Integration (CI). Each unit combines the data from multiple antennae into a single output in the complex domain or power domain, depending on how it is configured. When an Integration unit is enabled the Input DMA Engine should also be configured in Integration Mode (RM\_CONF2.PM=IM) so that the buffer RAM contains one FFT result from each connected antenna.

- 
- 1) 10 bits are stored to specify the fractional part of the significand with an implicit lead bit. The lead bit value is 1 unless the exponent field is stored with all zeros
  - 2) Support for subnormal numbers is not implemented. Any conversion result which would require representation using a subnormal number is instead represented as 0.
  - 3) With fractional component 65535.99997

## Signal Processing Unit 2 (SPU2)

The units are enabled and their operation configured by setting NDCTRL.MODE while writing of the output to Radar Memory is enabled by ND\_BASEb.EN. If a unit is not enabled then its related ND\_BASEb.EN has no effect. The antennae included in the sum of products are selected by a bit mask in NDCTRL.USEANTI and each selected antenna has its own multiplication constant. The summed result can be scaled to prevent overflow. The scaling operation is configured using the NDCTRL.SCALEi bitfield. The format of the output data is controlled by the NDCTRL.FORMATi bitfield. Data can be written as 16 bit, 32 bit integer or half precision floating point real or complex values. For the real-only data output modes when the Unit is in DBF mode, the power ( $|•|^2$ ) of the complex value is calculated before output to memory. The output that connects to the CFAR engines is always the power value.

Note that, if either Integration module is enabled, the CFAR module can only be used to process output from one of the Integration units, as reading data for direct input to the CFAR and Integration require different addressing modes for the Buffer RAM. The CFARCTRL.SRCSEL field should be set to INT0 or INT1 when the Input DMA Engine is configured in Integration Mode.

**Table 812 Integration Unit Configuration Parameters**

Parameter	Definition	Comments
Unit operation mode	enable or disable each unit and set its operation mode <sup>1)</sup>	<b>NDCTRL.MODE</b>
operand output format	16 or 32bits, half precision floating point	<b>NDCTRL.FORMATi</b>
weights	individual weights per antenna stored in configuration memory	32bit precision <b>NDCBASE.NDCBASEi</b>
antennae	which antennae to include	<b>NDCTRL.USEANTI</b>
result rescaling	optional result division by 1 or by 2 or by 4 or by 8.implemented using shift where sign extension is maintained.	to avoid saturation when having multiple antenna <b>NDCTRL.SCALEi</b>
output base address	starting point in memory where data shall be written	<b>ND_BASEi.BASE</b>
output enable	enable output to Radar Memory	<b>ND_BASEi.EN</b>

- 1) If the integrated data will neither be used by other processing units nor written to Radar Memory then enabling this bit will have no effect on processing flow i.e. the extra read pass through the buffer memory will not take place.

**Table 813 Integration Unit Modes**

Parameter Value	Unit 0 Operating Mode	Unit 1 Operating Mode
<b>OFF</b>	Disabled	Disabled
<b>NCI_OFF</b>	NCI	Disabled
<b>NCI_NCI</b>	NCI	NCI
<b>NCI_DBF</b>	NCI	DBF
<b>DBF_OFF</b>	DBF	Disabled
<b>DBF_DBF</b>	DBF	DBF

### 20.3.5.4.1 Digital Beam Forming

A unit performs DBF by combining the equivalent values from each antenna stored in the Buffer RAM using a complex multiplication of the sample point followed an addition. Each antenna can have its own multiplication constant. The multiplication weights are 16 bit, signed, complex numbers.

## Signal Processing Unit 2 (SPU2)

The input to DBF is 32 bit signed complex data. This is multiplied by the 16 bit complex weighting constant to give a 48 bit result. The resulting value is added into two 51 bit accumulators, one for each component of the complex data. Once the values derived from the equivalent bins of all FFT results in the buffer RAM are added into the accumulators, the accumulator results can then be right shifted by 0, 1, 2 or 3 bits depending on the NDCTRL.SCALEi setting.

The 32 bit output value for each component will be taken from bits [47:16] of the relevant accumulator value. The operation is equivalent to dividing by  $2^{15}$  and then rounding down to the nearest integer.

For power format output, the power of the complex result is calculated using the method described in [Section 20.3.5.1.1, “Linear Power Calculation” on Page 40](#).

If a 16 bit precision result output is configured, each component will be derived from bits [47:32] of the relevant accumulator. The operation is equivalent to dividing the accumulator by  $2^{31}$  and rounding to the nearest integer.

For half precision floating point output the value will be taken from bits [47:16] as for the 32-bit value and then treated the same way as FFT data as described in [Section 20.3.5.3.4, “Half Precision Floating Point Format” on Page 48](#).

If any of bits [49:47] of the accumulator value are different to the sign bit (bit 50), then an overflow condition is detected and the output shall be saturated according to the value of the sign bit. For integer formats this is maximum positive and maximum negative. For half precision floating point formats the output value is set to the IEEE 754 Infinity value of  $7C00_H$  for positive sign and  $FC00_H$  for negative sign. This saturation is also reflected in the result of the power calculation except the saturated value is always positive.

Normally the NDCTRL.DBFREAL should be set to COMPLEX. Setting the NDCTRL.DBFREAL field to REAL will result in only the real part of the weights and FFT bin value being used in the multiplication and real result being fed directly to the addition. When NDCTRL.DBFREAL==REAL and a complex output format is selected the imaginary part shall be set to zero.

**Table 814 Digital Beam Forming Configuration Parameter**

Parameter	Definition	Comments
real mode	real-only calculation: ignore imaginary part, zero imaginary part of output	<b>NDCTRL.DBFREAL</b>

### 20.3.5.4.2 Non-Coherent Integration

The input to NCI is unsigned 32 bit linear power. If the data in the buffer memory is already a power value, due to the setting of RM\_CONF2.FORMAT, then the power calculation is bypassed. The input is multiplied by the 16 bit real part<sup>1)</sup> of the weighting constant to give a 48 bit result. This weight is unsigned. The weighted power value is added into a 51 bit accumulator. Once the power values derived from the equivalent bins of the selected FFT results in the buffer RAM are added into the accumulator, the accumulator result can then be right shifted by 0, 1, 2 or 3 bits depending on the NDCTRL.SCALEi setting. If the data in the buffer memory is complex then a 32 bit linear power value is calculated before the multiplication.

The 32 bit output value will be taken from bits [47:16] of the accumulator value. The operation is equivalent to dividing by  $2^{16}$  and then rounding down to the nearest integer.

If a 16 bit precision result is configured this will be derived from bits [47:32] of the accumulator. The operation is equivalent to dividing the accumulator by  $2^{32}$  and rounding to the nearest integer.

The half precision floating point value will be taken from bits [47:16], the same as for the 32-bit output value, and then treated the same way as FFT data as described in [Section 20.3.5.3.4, “Half Precision Floating Point Format” on Page 48](#).

1) The coefficients in configuration memory are still 16-bit complex values but the imaginary part is ignored when in NCI mode.

## Signal Processing Unit 2 (SPU2)

For the integer output formats, if any of bits [50:48] of the accumulator result are set, then an overflow condition is detected and all of bits of the output will be set to saturate the output value. For half precision floating point formats the output value is set to the IEEE 754 +Infinity value  $7c00_{H}$ .

### 20.3.5.5 Constant False Alarm Rate Module

The CFAR module is used to identify interesting points in the data and then filter the data so that only these points are written to Radar Memory. There are 2 engines that may operate at the same time.

For each engine, if output is enabled, the results are written to Radar Memory as a table. The same information is also always fed to the Bin Rejection unit via the Logical Function. The output of that Logical Function may also be written to Memory as a list of labels that locate each point of interest in the 3D space formed of the bin number, FFT number and antenna number.

#### 20.3.5.5.1 CFAR Input Sources

There are 3 selectable sources of data for the CFAR module input as described below. The data source is selected by the CFARCTRL.SRCSEL field. The same input data goes to both engines.

##### FFT Data

Each ramp from each antennae is used to generate a set of sample power values taken from the unit described in [Section 20.3.5.1.1, “Linear Power Calculation” on Page 40](#).

##### Data from Integration Unit 0

As the data is read from the memory, it is routed through the first Integration Unit to produce a composite sample power value. This is then used for the CFAR input.

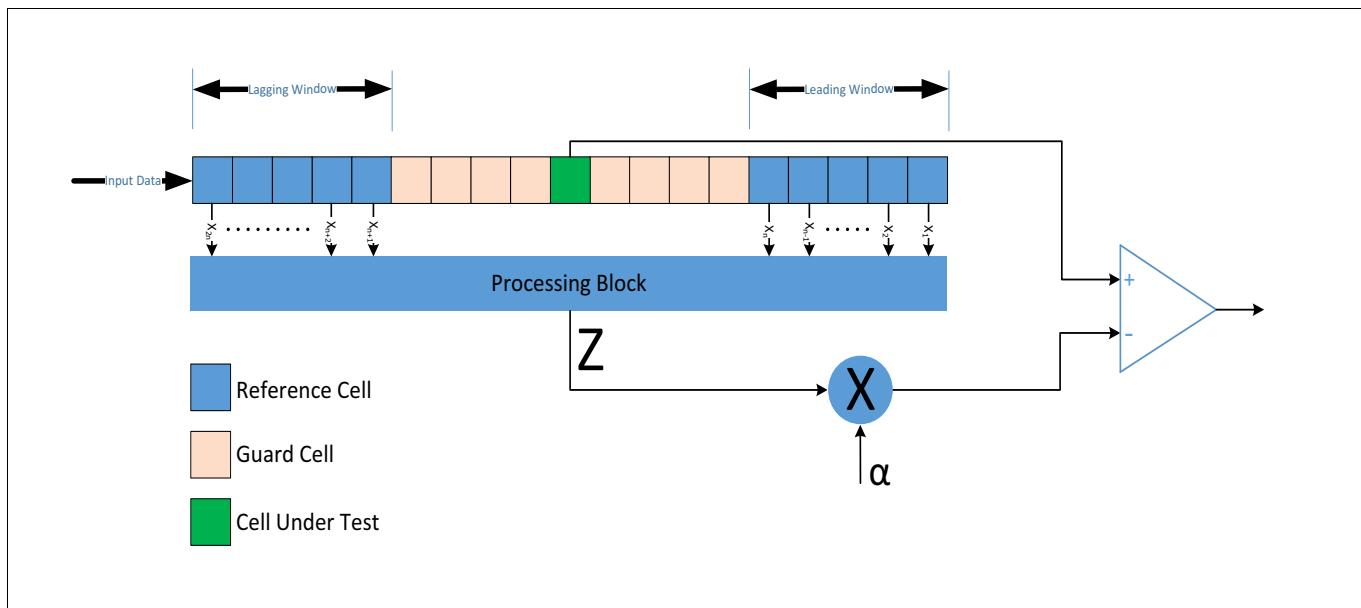
##### Data from Integration Unit 1

As the data is read from the memory, it is routed through the second Integration Unit to produce a composite sample power value. This is then used for the CFAR input.

#### 20.3.5.5.2 CFAR Engine Architecture

The following diagram shows the top level architecture of the CFAR engine

## Signal Processing Unit 2 (SPU2)



**Figure 262 CFAR Engine Architecture**

Input power samples are shifted through the processing window, which consists of windows of reference cells on either side of the Cell Under Test (CUT). Each window is separated from the Cell Under Test by a configurable number of guard cells. The windows cells are processed to generate a threshold value ( $Z$ ). This is then scaled for comparison with the CUT value. Note that in this diagram, the threshold is shown as  $Z$  multiplied by a constant "alpha". The actual implementation outputs  $Z$  as  $\log_2$  power, not linear power so the actual implementation is  $\log_2(Z) + \text{beta}$ , where beta is  $\log_2(\alpha)$ . The parameter beta is available as a register bitfield for both engines.

### 20.3.5.5.3 CFAR Engine Data Format

The CFAR engine uses a 64 bit linear power value internally. This is obtained from the 32 bit values computed by the TC3Ax SPU logic by multiplying by  $2^{31}$  to generate a maximum possible value of  $7FFF\_FFF\_8000\_0000_H$ . Where the CFAR engine uses  $\log_2$  power for internal calculations or comparison, this is computed as a 16 bit log2 value with 7 integer and 9 fractional bits from the 64 bit linear power value.

### 20.3.5.5.4 CFAR Module Configuration

The CFAR module contains two, independent engines. One engine implements the CA-CFAR algorithm and some derivatives (including CASH-CFAR). The other implements the GOS-CFAR algorithm and some derivatives. Both engines can be independently enabled and configured.

The engines are controlled by the CFARCTRL.CFAR\_CAE bitfield for the CA engine and the CFARCTRL.CFAR\_GOSE bitfield for the GOS engine. The results from one or both engines may be written to Radar Memory by setting the CABASE.EN or GOSBASE.EN bitfields. Separate base addresses for output are defined by the CABASE.BASE and GOSBASE.BASE bitfields. If an engine is not enabled then its output enable (EN) bitfield has no effect and no data will be written to Memory.

The CFAR results can also be used to control the Bin Rejection unit. This is enabled by setting the CFARCTRL.LFUNC bitfield to a value that results in the output of the Logical Function being influenced by one or both engines. When an engine is not enabled its output to the Bin Rejection unit is fixed to zero.

The results data will be written to Radar Memory at incrementing contiguous addresses starting from the programmed base address for each engine. Results will be in ascending bin order with bin 0 stored at the LSB of the word at the lowest address. Results of less than 256 bits in length will be aligned on a power of 2 bit address boundary i.e.

## Signal Processing Unit 2 (SPU2)

- results of more than 8 bits size and less than 17 bits size will be aligned on 16 bit boundaries (one result in every 2 bytes of memory) results of more than 16 bits size and less than 33 bits size will be aligned on 32 bit boundaries (one result in every 4 bytes of memory)
- results of more than 32 bits size and less than 65 bits size will be aligned on 64 bit boundaries (one result in every 8 bytes of memory)
- results of more than 64 bits size and less than 128 bits size will be aligned on 128 bit boundaries (one result in every 16 bytes of memory)
- results of more than 128 bits size and less than 257 bits size will be aligned on 256 bit boundaries (one result in every 32 bytes of memory)
- Results of more than 256 bits will be rounded up in size to the next 256 bit (32 byte) address boundary.
  - e.g. a result with 734 bits of data will be fitted into 768 bits (96 bytes) of memory

**Table 815 1D-CFAR Engines Common Configuration Parameters**

Parameter	Definition	Comments
CA Base Address	Starting address for writing CA-CFAR results to memory	<b>CABASE.BASE</b> Word Address, must be 256 bit (32 byte) aligned
GOS Base Address	Starting address for writing GOS-CFAR results to memory	<b>GOSBASE.BASE</b> Word Address, must be 256 bit (32 byte) aligned
Number of bins	From FFT8 to FFT2048. Any intermediate power of 2 value supported	FFT size is passed from the loader configuration <sup>1)</sup> <b>LDR_CONF.SIZE</b>
CFAR input selection	Select input from FFT engine, INT0 or INT1	Note that when input from FFT engine is used, signal power will be computed on each received BIN. <b>CFARCTRL.SRCSEL</b>
Spectrum extension mode	Select one of two extension modes depending on whether CFAR is run on range or doppler: mode1 = spectrum extension for range mode2 = spectrum extension for doppler	<b>CFARCTRL.EXTNSEN</b>
Spectrum Extension Window	Number of extra cells to be added on each end of the CFAR data	<b>CFARCTRL.SEWIN</b>
CFAR enable	CFAR enabling / disabling	<b>CFARCTRL.CFAR_CAE and CFARCTRL.CFAR_GOSE</b>
Label mode select	Enable and select label list output mode	<b>CFARCTRLLBLMODE</b>

1) If the FFT accelerator is bypassed, the size of the input data is derived from the bin loop repeat value (ID\_RM\_BLR.BLR) if data is sourced from Radar Memory or sample count (ID\_CONF.SMPLCNT) if the data is sourced from the RIF

### 20.3.5.5.5 GOS-CFAR Engine

Configuration Parameters for the GOS-CFAR Engine are detailed in the following table

## Signal Processing Unit 2 (SPU2)

**Table 816 GOS-CFAR engine SW configuration parameters**

Parameter	Definition	Comments
GOS-CFAR algorithm	Selects between CA, GO and SO	<b>CFARCFG1.GOSALGO</b>
number of cells in leading and lagging windows for GOS-CFAR	Can be set to any value from 1 to 32 cells either side of the cell under test	leading and lagging window <b>CFARCFG2.GOSWINCELL</b>
number of guard cells	This field has a possible value range of 0-63 but only values from 0 o 30 are permitted.	<b>CFARCFG2.GOSGUARD</b>
GOS-CFAR Beta parameter	16 bit, unsigned number with 7 integer bits and 9 fractional bits. Used to adjust the threshold i.e. Cell Under Test must exceed the trigger value by more than this parameter. The parameter represents $\log_2$ power	<b>CFARCFG3.GOSBETA</b>
GOS-CFAR Lead Index	Index of sorted statistic in leading window in GOS-CFAR	<b>CFARCFG2.IDXL0</b> min value 0, max value GOSWINCELL-1
GOS-CFAR Trailing Index	Index of sorted statistic in lagging window in GOS-CFAR	<b>CFARCFG2.IDXLG</b> min value 0, max value GOSWINCELL-1

In the GOS-CFAR algorithms, the samples within the leading/lagging windows are processed through a non-linear sorting operation and selected ordered statistics:  $Y_1$ ,  $Y_2$  are extracted from the two sub-windows, respectively. The GOSCA, GOSGO, and GOSSO algorithms differ in the manner  $Y_1$ ,  $Y_2$  are combined:

- GOSCA:  $Z = Y_1 + Y_2$
- GOSGO:  $Z = \max\{Y_1, Y_2\}$
- GOSSO:  $Z = \min\{Y_1, Y_2\}$

The resulting statistic  $Z$  is finally scaled by the “beta” scaling factor in order to determine the threshold value. The value of the “beta” parameter determines the probabilities of False Alarm (FA) and Missed Detection (MD). For the GOSGO and GOSSO algorithms the  $\log_2$ -domain input power sample implementation is directly equivalent to the linear one, since both the sorting and max/min operations produce equivalent ( $\log_2$ -domain) values for  $Z$ . The only difference is that the threshold value is produced by scaling  $Z$  additively with “beta”. On the other hand a direct  $\log_2$ -domain implementation of the GOSCA algorithm is not normally equivalent to the linear-domain implementation, since  $\log_2(Y_1 + Y_2)$  is not equal to  $(\log_2(Y_1) + \log_2(Y_2))$ . However in the CFAR module an equivalent  $\log_2$ -domain result to the linear domain one is obtained by post processing.

### 20.3.5.6 CA-CFAR Engine

The CA-CFAR engine supports the CA-CFAR, CAGO-CFAR, CASO-CFAR and CASH-CFAR algorithms. The configuration Parameters for the CA-CFAR Engine are detailed in the following table:

## Signal Processing Unit 2 (SPU2)

**Table 817 CA-CFAR engine configuration parameters**

Parameter	Definition	Comments
CA-CFAR algorithm	selects between CA, CAGO,CASO and CASH	<b>CFARCFG1.CAALGO</b>
CA-CFAR Beta parameter	16 bit, unsigned number with 7 integer bits and 9 fractional bits. Used to adjust the threshold i.e. Cell Under Test must exceed the trigger value by more than this parameter. The parameter represents $\log_2$ power	<b>CFARCFG1.CABETA</b>
number of guard cells	can be set to any value from 0 to 16 <sup>1)</sup>	any value from 0 to 16 permitted <b>CFARCFG1.CAGUARD</b>
number of cells in leading and lagging windows for CA-CFAR	This defines the window size for comparison. The window size used is $2^n$ , where n is the value in the register bitfield	leading and lagging window <b>CFARCFG1.CAWINCELL</b> min value 1, max value 5
active cells in window to be used for averaging for CA-CFAR/CASH-CFAR	This defines the window size for averaging within the leading and lagging windows. The window size used is $2^n$ , where n is the value in the register bitfield	<b>CFARCFG2.CASHWIN</b> min value 0, max value 5. Must be less than or equal to CAWINCELL value when CASH algorithm is selected. Otherwise must be equal to CAWINCELL

1) Values greater than 16 will cause undefined operation and incorrect results

In the CASH and CA-CFAR algorithms, the samples within the leading/lagging windows are processed through sample averaging to produce  $Y_1$ ,  $Y_2$  statistics for the leading and lagging windows respectively. Averaging is performed in the linear power domain and conversion into the  $\log_2$ -domain is performed prior to the application of the non-linear/linear operation in the processing block as shown in [Figure 262 “CFAR Engine Architecture” on Page 52](#).

In CASH-CFAR, the  $\log_2$ -converted averaged power samples are processed through MAX and MIN operations, as described in “F. X. Hofele, An Innovative CFAR Algorithm, CIE Intern. Conf. on Radar, 2001” and associated patents.

In CA/CAGO/CASO-CFAR the  $\log_2$ -domain processing is similar to the GOS-CFAR.

Threshold scaling in CASH/CA-CFAR is performed additively using the “beta” parameter in a similar manner to that used by GOS-CFAR.

### 20.3.5.5.7 CFAR Engine Configuration Restrictions

The CFAR engine requires some constraints on configuration to avoid undefined results

- The value of two times the sum of the number of guard cells and the number of cell in the window must not exceed the FFT size minus one.
- If spectrum extension is enabled, the spectrum extension window size, CFARCTRL.SEWIN, must equal to the largest sum of the number of guard cells and the number of window cells for all CFAR engines that are enabled. The sum must be calculated for each engine and the largest value must be used.
- The achievable maximum value for the  $\log_2$  power representation of the linear power is  $7DFF_H$  (for a linear power of  $7FFF\_FFFF\_8000\_0000_H$ ) If the Beta parameter exceeds this value, it is impossible for any detection to occur.

---

## Signal Processing Unit 2 (SPU2)

### 20.3.5.5.8 CFAR spectrum extension

Two extension modes need to be considered: range extension and velocity extension.

Range and Velocity extension is supported by shifting additional data points into the CFAR unit at the beginning and end of the FFT to preload the window cells with the correct data. This requires the equivalent number of output data points from the CFAR to be ignored. This will be handled transparently by the supporting logic if this mode is enabled

Spectrum extension is always done per FFT dataset by adding additional bins to the dataset. The content replicating the value of some of the existing bins into the new bins. Data is fed into the CFAR unit on a “per antenna” basis. Spectrum Extension never uses data from more than one antenna. The bins to be replicated depend on the spectrum extension mode.

Spectrum extension will cause some data to be read twice from the buffer memory. The amount of extra reads will be directly linked to the size of the extension defined by CFARCTRL.SEWIN. SEWIN should be set to be no larger than the largest window as required according to the settings of the active units: CA CFAR, GOS CFAR and LCLMAX.

**Note:** *Spectrum Extension is used in conjunction with the CFAR or Local MAX functions and should not be enabled if neither of these functions is enabled. Enabling the Spectrum Extension in the absence of enabling either CFAR or Local Max is not supported and will result in abnormal operation of the TC3Ax SPU.*

#### Spectrum Extension for CFAR in Range

In range spectrum extension, the spectrum is extended by mirroring across the border as illustrated in the figure below. In this figure, the y-axis is along the FFT dataset (bin number ascending from the bottom of the figure).

## Signal Processing Unit 2 (SPU2)

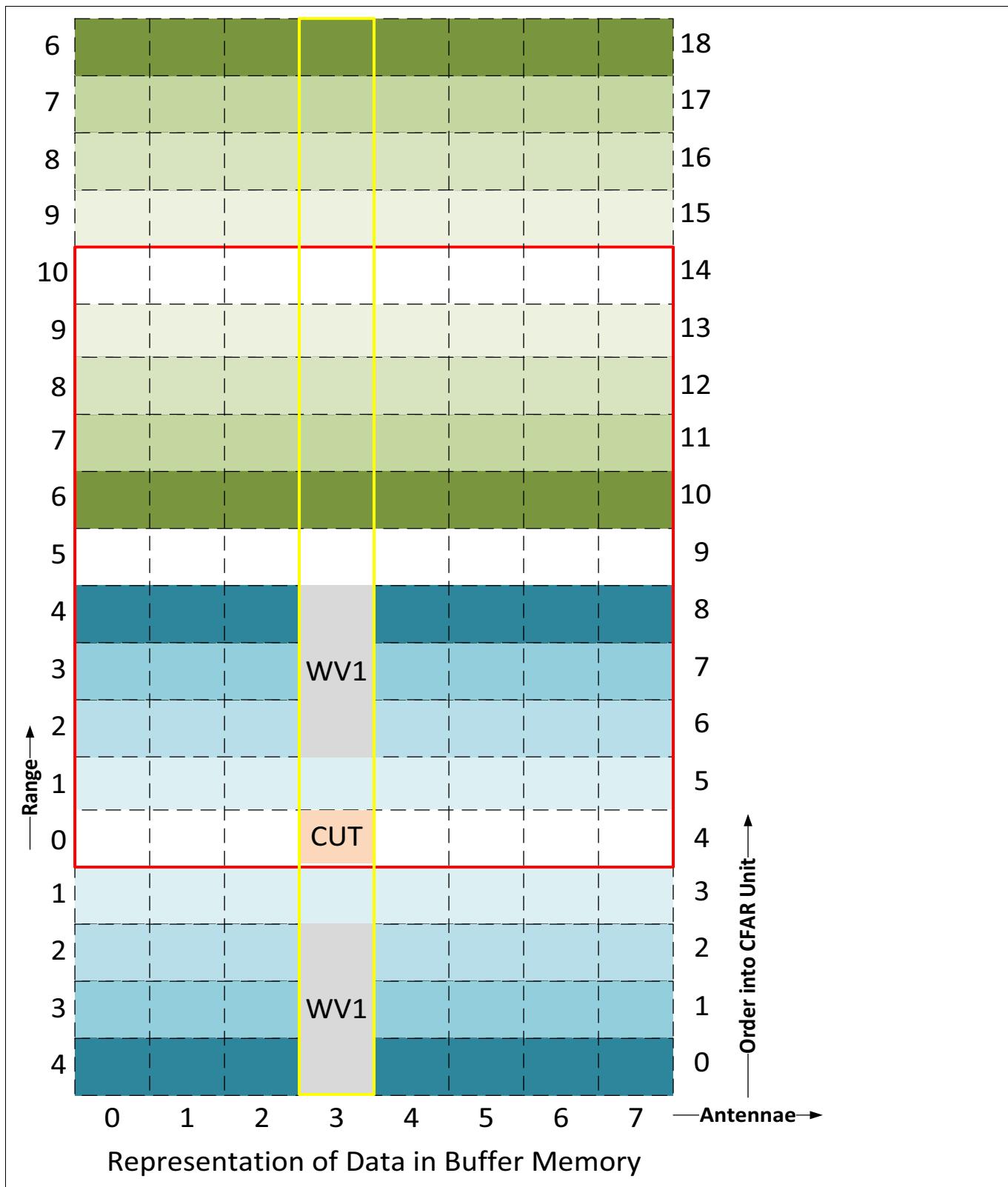


Figure 263 Diagram of Range Spectrum Extension

Please note that the figure is only showing the principle as real extension needs to be done according to the window size for range and for velocity directions. The extent of the “real” data is shown by the red rectangle. The effective size of the dataset and the mirroring used to achieve it is shown by the color coding. The extended

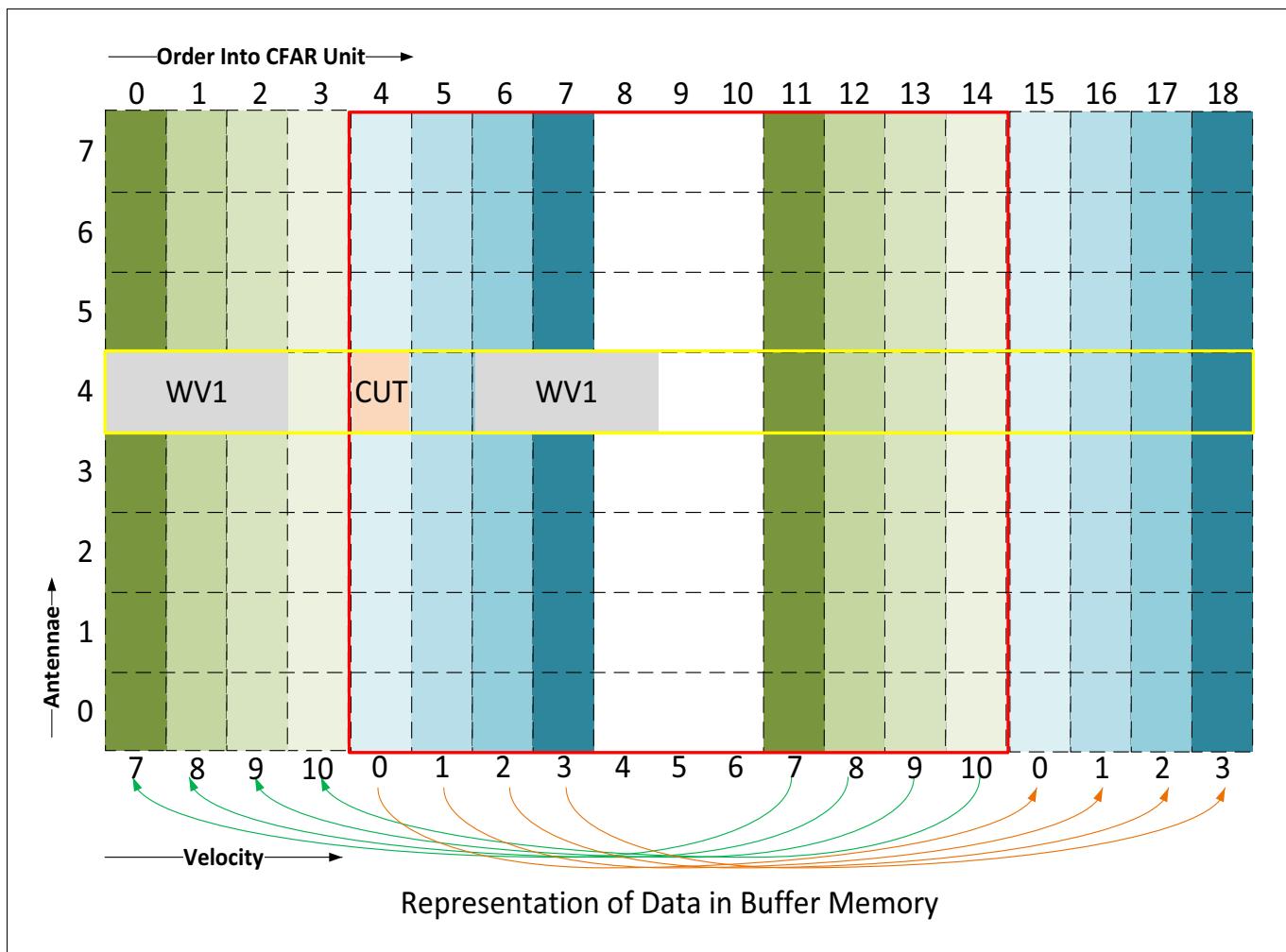
## Signal Processing Unit 2 (SPU2)

dataset fed to the CFAR units is shown as the yellow rectangle. CUT is “Cell Under Test” and WV1 represents an example window with guard band.

The behaviour if the window size exceeds the size of the dataset is not defined.

### Spectrum Extension for CFAR in Velocity

In velocity / Doppler direction, the spectrum is extended by implementing cyclic repetition. The color coding in the diagram below shows how the repetition is implemented. The extent of the “real” data is shown by the red rectangle. Please note that the figure is only showing the principle as real extension needs to be done according to the window size for range and for velocity directions.



**Figure 264 Diagram of Velocity Spectrum Extension**

The extent of the “real” data is shown by the red rectangle. The effective size of the dataset and the mirroring used to achieve it is shown by the color coding. The extended dataset fed to the CFAR units is shown as the yellow rectangle. CUT is “Cell Under Test” and WV1 represents an example window with guard band.

The behaviour if the window size exceeds the size of the dataset is not defined.

## Signal Processing Unit 2 (SPU2)

### Spectrum Extension on Outputs from Integration (NCI/DBF)

Spectrum extension is always a one-dimensional operation which affects the input to the CFAR unit. When using spectrum extension with input from the integration units, the integration is performed first and the spectrum extension is then applied to the resulting one-dimensional vector as if dealing with a single antenna system.

### Spectrum Extension for Local Maximum

Spectrum extension for the Local Maximum function operates in the same way as for CFAR. When both Local Maximum and CFAR are enabled then the extension window should be set to suit the largest of the values need by the CFAR and Local Maximum units.

#### 20.3.5.5.9 Operation of Spectrum Extension

The spectrum extension modes work by adding extrapolated bins to each end of the FFT results. The number of pins added depend on the window size in use. For a window of e.g. 7 bins on each side of the bin under evaluation and additional 7 extrapolated bins will be added to each end of the FFT data to ensure that the first and last real bins are evaluated against a full set of window data.

Operation of the spectrum extension modes is controlled by the bitfield CFARCTRL.EXTNSN

For range spectrum extension, the extrapolated bins are mirrored. This means that when bin 0 is under comparison, real bins 1 to n are used for the values of extrapolated bins -1 to -n respectively (where n is determined by the comparison window size). For a 256 point FFT real bins 254 to 254-n are used for the values of extrapolated bins 256 to 256+n when evaluating bin 255.

For velocity spectrum extension, the extrapolated bins are determined by cyclic repetition. Again, using a 256 bin FFT as an example, when evaluating bin 0, extrapolated bins -n to -1 will be the values of real bins 255-n to 255. When evaluating bin 255, extrapolated bins 256 to 256+n will be the values of real bins 0 to n.

#### 20.3.5.6 Statistical Information

The output of Statistical data is enabled by setting SBCTRL.EN to 1<sub>B</sub>.

The mean value computed by the Unloader Unit is queued for writing to the Radar Memory. Additionally, the mean value is used to compute the variance (standard deviation squared) for each FFT. While the FFT data is being parsed to compute the variance, the minimum and maximum power values will be captured

Along with the minimum and maximum power values, the index of the FFT bin at which the minimum and maximum occurred will also be captured.

In the event of multiple bins having the same minimum or maximum, only the index of the first bin to have the value will be captured.

The base address for the writes to start is defined in SBCTRL.BASE. and configuring either of the enable conditions will result in the statistical information output channel being enabled. However, only the requested data will be written. When SBCTRL.EN is 1 then the statistical information will be written in packets of 32 bytes per FFT (256 bit aligned) with 20 bytes of data and 12 spare bytes of indeterminate value. See [Table 818 "Statistical Information Written" on Page 59](#) for the data format used.

**Table 818 Statistical Information Written**

Parameter	Size (bytes)	Address Offset
Minimum Power	4 <sub>D</sub>	0 <sub>D</sub>
Maximum Power	4 <sub>D</sub>	4 <sub>D</sub>
Mean Power	4 <sub>D</sub>	8 <sub>D</sub>

## Signal Processing Unit 2 (SPU2)

**Table 818 Statistical Information Written (cont'd)**

Parameter	Size (bytes)	Address Offset
Variance	4 <sub>D</sub>	12 <sub>D</sub>
FFT Bin Index for Minimum Power	2 <sub>D</sub>	16 <sub>D</sub>
FFT Bin Index for Maximum Power	2 <sub>D</sub>	18 <sub>D</sub>

The configuration parameters affecting the statistical data output are shown in [Table 819 “Statistical Unit Configuration Parameters” on Page 60](#)

**Table 819 Statistical Unit Configuration Parameters**

Parameter	Definition	Comments
statistical data enable	enable / disable	<b>SBCTRL.EN</b>

### 20.3.6 Output DMA Engine

This section describes the operation of the DMA engine that the TC3Ax SPU uses to write results to memory.

#### 20.3.6.1 Output DMA Engine Channels

As described in [Table 804 “Output DMA Engine Data Streams” on Page 38](#), the Output DMA Engine has nine independent data channels. These are configured using the following parameters:

**Table 820 Output DMA Engine Configuration Parameters**

Parameter	Definition	Comments
DMA base address for port1	DMA base address when starting a new FFT sequence from Radar memory	FFT results only <b>ODP_BASE.BASE</b>
DMA inner loop address offset	Offset address to be added to base address	FFT results only
DMA outer loop address offset	Offset address to be added to base address	These parameters are used for in-place FFT
DMA inner loop repeat value	Number of times of inner loop execution	
DMA outer loop repeat value	Number of times of butter loop execution	
DMA base address for port2	Signal power	<b>PWRCTRL.BASE</b>
DMA base address for port3	Statistics: min, max, mean, variance	<b>SBCTRL.BASE</b>
DMA base address for port4	Thresholding with CFAR CA	<b>CABASE.BASE</b>
DMA base address for port5	Thresholding with CFAR GOS	<b>GOSBASE.BASE</b>
DMA base address for port6	Non-Coherent Integration or Digital Beam Forming	<b>NDBASE0.BASE</b>
DMA base address for port7	Non-Coherent Integration or Digital Beam Forming	<b>NDBASE1.BASE</b>
DMA Base Address for port8	Thresholding with Local Maximum or Logical Function	<b>LCLMAXBASE.BASE</b> Output selected with CFARCTRL.LFOSEL
DMA Base Address for port9	Label list output	<b>LBLBASE.BASE</b>

## Signal Processing Unit 2 (SPU2)

**Table 820 Output DMA Engine Configuration Parameters (cont'd)**

Parameter	Definition	Comments
signed FFT BIN casting/rescaling	defines how many bits are kept from the LSB (of the 32bit internal precision) note that the sign, located on the MSB, is maintained.	<b>ODP_CONF.EXPNT</b>
output FIFO flush <sup>1)</sup>	FIFOs are flushed upon each configuration change. Required for the last computation step	This is an implicit parameter triggered internally when the processing of the last FFT in the measurement cycle is completed

1) Automatic Operation, no specific control bit included for this

Each of the BASE fields defines the start address for the output channel. The write address will be incremented automatically so that the results of each channel are stored as an array in memory starting from the defined base address.

The exception is the “In Place FFT” mode for writing the FFT results. Here the ODP\_BASE register is used as the start address and the address sequence is calculated using the Output DMA Engine Loop Count and Loop Offset parameters in the ODP\_ILO, ODP\_OLO, ODP\_BLO and ODP\_IOLR registers which behave in the same way as the equivalent registers in the Input DMA (see “[In Place FFT](#) on Page 36 and “[Load from Radar Memory](#) on Page 5).

The output data stream is pushed to Radar Memory without any possibility of delay or wait states. The Radar Memory is required to keep pace with the output data rate. This is achieved by giving writes absolute priority in arbitration and arranging separate arbitration for each 256 KiB Radar Memory tile.

**Note:** *In systems with two active SPUs or with BYPASS\_CTRL.BYPASS enabled, it is mandatory to ensure that the output from each TC3Ax SPU is directed to different tiles of the Radar Memory. In the event of arbitration being required between two TC3Ax SPU writes, then the write which loses arbitration will be lost.*

**Note:** *The TC3Ax SPU is specified to cope with Radar Memory sizes up to 16 MiB. Any accesses outside the range of the Radar Memory on the device will result in an error being generated by the Radar Memory module.*

### 20.3.6.2 Data Cube Organisation and Size after processing ADC data

The FFT results of processing ADC sample sets (ramp data) are always kept together in memory.

Each FFT result always starts at a 256 bit (32 byte) aligned address. This is mandatory if the data is to be read back into the TC3Ax SPU as the read operations controlled by the Input Data Manager require the start address of each dataset to be 256 bit aligned.

The ramp data are then grouped by antennae so the ramp n<sup>1)</sup> data for all antennae (inner loop output) is grouped together in memory. The size of the data cube is variable and depends on the following factors

- data format, real or complex
- data precision 16 bit or 32 bit (DPR)
- samples per ramp or FFT output (maximum bin loop count, BLC)

1) where n is the ramp number and also the outer loop count

## Signal Processing Unit 2 (SPU2)

- number of antennae (NA)
- number of ramps in measurement cycle (maximum outer loop count, OLC)

The size of each data point (SDP) is as follows

- 16 bit real: 2 bytes (DPR = 16 bit)
- 32 bit real: 4 bytes (DPR = 32bit)
- 16 bit complex: 4 bytes (DPR = 16 bit)
- 32 bit complex: 8 bytes (DPR = 32bit)

The size in bytes of the data generated for each configured antennae per ramp (ADS) is

$$\text{ADS(bytes)} = \text{BLC} \times \text{SDP} \quad (20.5)$$

The value for ADS will always be rounded up to the next 32 bytes (ADSR).

The overall dataset size (DS) is then

$$\text{DS(bytes)} = (\text{ADSR}) \times \text{NA} \times \text{OLC} \quad (20.6)$$

As the maximum amount of data that can fit into the Buffer RAM is 64kBytes, this gives a hard limit for the amount of data per inner loop (the ADS value).

### 20.3.7 Radar sequencer

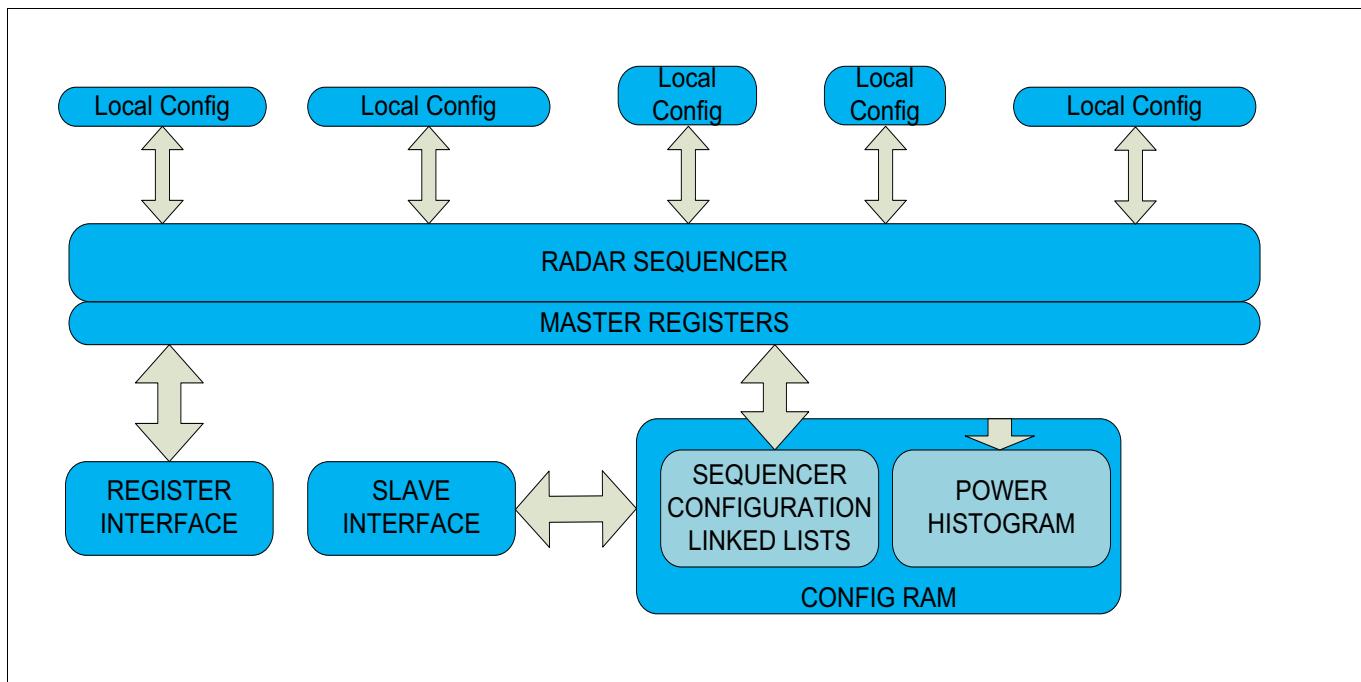
The Radar sequencer is the main control sequencer of the TC3Ax SPU. It manages the configuration/reconfiguration of each unit and controls the sequencing of operations. It also monitors each unit for correct operation and generates suitable signalling to the application software via the Interrupt system and the Safety Management Unit.

General principles of operation are:

- Configurations can be written directly to the registers of the TC3Ax SPU by application software
- However, configurations can also be stored in the configuration memory, which is a RAM accessible by the application software (this RAM is also used to store the histogram)
- Configurations may be organized in linked lists.

This following sections describe the features and behaviour controlled by the Radar Sequencer.

## Signal Processing Unit 2 (SPU2)



**Figure 265 Radar Sequencer Overview**

### 20.3.7.1 General Configuration

The Radar sequencer has several configuration options that define the overall configuration of the TC3Ax SPU.

**Table 821 General Configuration Options**

Function	Register	Bitfield	Comment
Clocking Mode	AUXCTRL	DIVF	Fractional clock division ratio for the LOADER/MATH0/MATH1/FFT/UNLOADER pipeline. The ratio is given by the formula $1 - DIVF/16$
	AUXCTRL	CLKRAMP	If enabled, the first and last input Buffer Memory of the configuration shall be processed at 50% of the TC3Ax SPU clock frequency
	CLC	DISR	Clock Disable. Switch off the TC3Ax SPU's internal clock
Linked List Base Address	CTRL	NXT_CONF	Base address for the next set of configuration information in the linked list
Start	CTRL	TRIG	Starts the Radar Sequencer if MODE==SW
Operating Mode	CTRL	MODE	Defines when the TC3Ax SPU will start processing input data
Attention	CTRL	ATTN	Trigger a service request interrupt when execution of the current configuration is completed
Cross Trigger	CTRL	XTRIG	Trigger a "DONE" event when execution of the current configuration is completed.

## Signal Processing Unit 2 (SPU2)

**Table 821 General Configuration Options (cont'd)**

Function	Register	Bitfield	Comment
Incremental Trigger	CTRL	ITRIG	Generate a pulse on the incremental trigger output after MATH2 has completed processing Buffer Memory contents and ODM has written all data to Memory.
Bypass Trigger	CTRL	BPTRIG	If set the Bypass function shall be triggered depending on the MODE setting
Last Configuration	CTRL	LAST	Current Configuration is the last configuration to be executed. Do not load a further configuration from the address defined by NXT_CONF

### 20.3.7.2 Radar sequencer start / stop

CTRL.MODE is used to define the conditions under which the TC3Ax SPU will start processing data.

The register space is organised so that the CTRL register should be the last register read during configuration by linked list. This ensures that a software trigger always starts with a valid data set. Starting processing of a linked list can be started either by writing a full set of register values directly into the register site finishing with a valid write to the CTRL register or by writing a MODE value of “RELOAD” and a valid NXT\_CONT value to the CTRL register

Writing a 1<sub>B</sub> to the CTRL.TRIG bit starts the radar sequencer provided that the CTRL.MODE bitfield is set to “SW” (software).

The TC3Ax SPU will stop processing a linked list when processing completes on a configuration which has the CTRL.LAST bit set or if the configuration just loaded contains “OFF” or “STOP” in the CTRL.MODE field.

**Note:** *The only difference between setting CTRL.LAST and loading a configuration set with CTRL.MODE set to OFF is that using CTRL.LAST eliminates an unnecessary configuration load. CTRL.MODE=STOP should only be used if it is required to abort processing of a linked list. If it is loaded from configuration memory then it will behave in the same way as setting CTRL.MODE to OFF.*

The CTRL.BUSY bitfield shows if the TC3Ax SPU is running or is stopped. Loading a configuration which has a CTRL.MODE field not set to OFF or STOP will cause the TC3Ax SPU to transition to the running state even if it is then just waiting for a trigger event. Once in the running state, software can only stop the TC3Ax SPU by writing “STOP” to CTRL.MODE. Any other value will have no effect.

**Note:** *Only the CTRL register should be written while the TC3Ax SPU is busy. Writing any other register will have immediate effects on the TC3Ax SPU configuration and may lead to undesirable consequences including a lock up of the TC3Ax SPU.*

**Table 822 CTRL.MODE Value Definitions**

Value	Function
OFF	TC3Ax SPU will do nothing. This is used to terminate a linked list operation when loaded from a configuration set in config RAM. Software writing OFF to the CTRL register once the TC3Ax SPU has been configured has no effect.
INT	TC3Ax SPU will pause waiting for data on the RIF input. Valid data on the RIF input will trigger operation. This can be qualified by the Partial Acquisition counter LIMIT value if the PACTR.TRIG bit is set

## Signal Processing Unit 2 (SPU2)

**Table 822 CTRL.MODE Value Definitions (cont'd)**

Value	Function
EXT	TC3Ax SPU will pause waiting for a trigger from an external source. Valid data on the RIF input will be ignored until the trigger event occurs. Valid data on the RIF data after the configured measurement cycle has completed will be ignored. If the Radar memory is configured as the data source, processing will commence immediately.
SPU0	TC3Ax SPU will trigger on an “DONE” event from SPU0 <sup>1)</sup>
SPU1	TC3Ax SPU will trigger on an “DONE” event from SPU1 <sup>1)</sup>
RELOAD	TC3Ax SPU will trigger an immediate configuration load from the value in the CTRL.NXTCONF field. Bypass will be triggered if CTRL.BPTRIG is set
SW	TC3Ax SPU will immediately start processing when a $1_B$ is detected in the CTRL.TRIG bitfield. The trigger event may occur simultaneously with “SW” being written to CTRL.MODE
STOP	This value should be written to the MODE bitfield if it is required to stop the TC3Ax SPU in the middle of linked list processing. If written while the TC3Ax SPU is processing data, the TC3Ax SPU will stop when the current processing operation is completed rather than loading the next configuration set. This value should not be loaded from a configuration set in RAM. In this case OFF should be used instead

1) Note that the TC3Ax SPU will only generate the “DONE” event when explicitly configured using the CTRL.XTRIG bitfield

The values INT and EXT are intended to be used when data is being read from one or more RIF instances.

The value EXT can also be used when reading data from Radar Memory.

SPU0, SPU1 and SW are intended to be used when reading data from Radar Memory.

As soon as any of the values: INT, EXT, SPU0, SPU1 or SW are written to the CTRL.MODE bitfield, the TC3Ax SPU will report its state as “BUSY” when CTRL.BUSY is read.

### 20.3.7.3 Synchronized Radar sequencer Start

The synchronised start function allows the 2 SPUs to be started at the same time. When using synchronised start, there will be no skew between the start point of the two SPUs. Synchronised start is configured setting the CTRL.MODE bitfield to EXT and triggering the start with an external source.

### 20.3.7.4 Configuration / Reconfiguration

During linked lists of computations, the Radar TC3Ax SPU supports full reconfiguration of the computation units of the TC3Ax SPU by reading the full register set from configuration memory. Partial reconfiguration is supported by making a copy of the previous configuration into the configuration memory and changing only the register fields affecting the functionality requiring reconfiguration before reloading it directly into the TC3Ax SPU registers using the linked list function.

### 20.3.7.5 Linked Lists Organization

A linked list is a set of configuration sets stored in Configuration Memory defining a sequence of computations that are run without CPU intervention. The linked list is stored in memory as an image of the registers, starting from ID\_CONF and ending at CTRL. The image of ID\_CONF is stored at the base address of the linked list entry and all the other registers are at the same relative offset as their address in the system memory map.

## Signal Processing Unit 2 (SPU2)

The registers outside the range from ID\_CONF to CTRL (including the CRC registers, CLC, MODID, STAT, MONITOR, OCS, ODA, USROTC, ACCEN, KRST0, KRST1 and KRSTCLR registers) cannot be reconfigured by the linked list function.

### 20.3.7.6 CPU monitoring during run time

The CPU(s) can read configuration registers during run time to check or wait for the radar sequencer to complete a computation sequence. The CPU can also read status registers to monitor the proper execution of the sequence. The following status registers can be monitored (read only) during run time

- ramp counter (derived from Input Data Manager)
- sample counter
- 1 busy flag per sub block
- 1 busy flag for TC3Ax SPU
- linked list sequence completed

### 20.3.7.7 Interrupts

The Radar TC3Ax SPU has a high degree of autonomy. As such, it generates few interrupts. Each interrupt source can be masked but for safety reasons, its flag can be read by application software.

Each TC3Ax SPU instance has two physical interrupt requests.

- Attention Request: asserted on one or more of events occurring during normal operation when intervention by the application software is needed
- Error Flag: asserted on an error condition when accurate completion of the calculation is compromised

The table below shows the different interrupt sources.

**Table 823 Interrupt Trigger Definition**

Event	Definition	Comments
End of execution	A configuration with the CTRL.ATTN bit set has completed execution	Attention Request <b>STATCTRL.INTMSK(0)</b>
Arithmetic Overflow During iFFT	The FFT accelerator has reported that an output number will not fit into the supported format	Attention Request <b>STATCTRL.INTMSK(1)</b>
Bypass Trigger	Set after the write of the last sample of the last ramp by Bypass function is confirmed.	<b>STATCTRL.INTMSK(2)</b>
End of linked list	Set after confirmed write of the last result of the outer loop.	Attention Request <b>STATCTRL.INTMSK(3)</b>
Partial Acquisition Counter Trigger	Partial Acquisition counter has reached programmed limit with PACTR.ATTN bit set	Attention Request <b>STATCTRL.INTMSK(4)</b>
Label List Counter Trigger 0	MD0_LBLCNT.CNT has reached half or full value as defined in CFARCFG3.LBLOLM	<b>STATCTRL.INTMSK(5)</b>
Label List Counter Trigger 1	MD1_LBLCNT.CNT has reached half or full value as defined in CFARCFG3.LBLOLM	<b>STATCTRL.INTMSK(6)</b>

## Signal Processing Unit 2 (SPU2)

**Table 823 Interrupt Trigger Definition (cont'd)**

Event	Definition	Comments
Radar Memory Read Error	Radar Memory has reported an error on read (see EMEM chapter for details)	Error Flag <b>STATCTRL.ERRMSK(0)</b>
Input DMA Unit write error	Address Overflow on Buffer Memory write	Error Flag <b>STATCTRL.ERRMSK(1)</b>
Radar Memory Write error	Radar Memory has reported an error on write (see EMEM chapter for details). This covers the Output DMA Engine, the Bypass Channel for RIF data and the MATH0 Output to Radar Memory	Error Flag <b>STATCTRL.ERRMSK(2)</b>
FIFO overrun	One or more of the TC3Ax SPU internal FIFOs has overflowed	Error Flag <b>STATCTRL.ERRMSK(3)</b>
Partial Acquisition Counter Error	Partial Acquisition Counter has reached limit with PACTR.ERR bit set	Error Flag <b>STATCTRL.ERRMSK(4)</b>
Input Data Overrun	The LOADER/FFT/MATH2 pipeline has not completed processing when the next set of FFTs have been fully written to buffer memory and the RIF has more data to be written for the following ramp. A ramp has (or ramps have) been skipped.	Error Flag <b>STATCTRL.ERRMSK(5)</b>

### 20.3.8 Streaming Processor 1, Buffer RAM Switching Behaviour

When the Input DMA Engine has finished writing a data block to the Buffer RAM, it will enable the switching of the Buffer RAMs to present the new data to the Loader module. If the Loader module is still reading the previous data block, the switch will not take place until the Loader module has finished.

The behaviour then varies depending on whether the data source is the Radar Interface or Radar Memory

#### 20.3.8.1 Data Source is Radar Interface

In the event that new data is presented at the Input DMA Engine inputs before the Loader has finished, the data will be written to the RAM and the switch of the buffer RAMs will not take place. This will be flagged as an "Input Data Overrun" error and can be used to trigger an TC3Ax SPU error interrupt (STATCTRL.ERRMSK(5)). As some applications may find controlled skipping of ramps useful, the maximum number of ramps skipped between buffer RAM switches occurring in a measurement cycle will be stored in the OVRRN field of the STAT register.

#### 20.3.8.2 Data Source is Radar Memory

Reading of further data from the Radar Memory will be stalled until the Loader has finished reading the previous data and the buffer RAM can be switched.

### 20.3.9 Configuration Memory

The configuration memory is used for several functions:

- Storing sets of configuration register settings for use by the linked list function of the TC3Ax SPU
- Storing Window Parameters for use by the FFT
- Storing the working data for the power histogram calculation

## Signal Processing Unit 2 (SPU2)

- Storing coefficients for integration (DBF and NCI)

The Configuration Memory is mapped into the system address space so that the application software can:

- write configuration register settings
- write window parameters and coefficients for integration (DBF and NCI)
- read power histogram information

Note that the operation of the histogram function requires all the available bandwidth of the RAM.

If the TC3Ax SPU is busy, then accesses to the configuration memory will be errored to prevent the bus access stalling for an extended period.

### 20.3.9.1 Safety/Security

As the configuration memory is primarily used for storing configuration settings, it is protected by the Access Enable mechanism used to prevent unauthorised modification of the TC3Ax SPU registers.

### 20.3.9.2 Configuration Register Data Format

The configuration register settings are stored in the configuration memory in the same organisation as they exist in the register space. Registers that can be updated from the configuration memory are in the address range ID\_CONF to CTRL. The ID\_CONF value will be stored at an address of  $0_H$  relative to the start address for the configuration set. The relative addresses of the data values for the other registers remain unchanged from their address map in system memory. The start address must be 64 bit aligned.

#### 20.3.9.2.1 Loading Configuration Settings

The loading of configuration settings into the registers can be triggered either manually or automatically by one of the following events:

- Software Trigger
- Trigger on TC3Ax SPU idle
  - This will trigger a complete register load when the last write is flushed to Radar Memory

### 20.3.9.3 Window Data Format

The window data is stored linearly in the configuration memory with the address offset corresponding to the data point that the window value should be used for. The address offset size is affected by the format of the window data set in LDR\_CONF2 and will vary between 2 and 8 bytes per coefficient. For complex window coefficients, the real component is stored at the lower address. An independent address offset can be defined for each antenna.

### 20.3.9.4 Configuration Memory Usage Restrictions

The configuration information is stored in a memory that is independent from the Radar memory. This memory is also used to store the results from the histogram unit and other parameters.

The histogram unit needs all the available bandwidth of this memory. The configuration memory is therefore organised as four independent banks. The use of the banks is allocated as shown in the following table.

## Signal Processing Unit 2 (SPU2)

**Table 824 Configuration Memory Bank Allocation**

	<b>Bank 1</b>	<b>Bank 2</b>	<b>Bank 3</b>	<b>Bank 4</b>
Address range	0 to CFG_RAM_SIZE/4-1	CFG_RAM_SIZE/4 to CFG_RAM_SIZE/2-1	CFG_RAM_SIZE/2 to 3*CFG_RAM_SIZE/4-1	3*CFG_RAM_SIZE/4 to CFG_RAM_SIZE-1
Register settings	Yes	Yes	Yes	Yes
Histogram data	Yes	Yes	No	No
Window parameters	Yes	Yes	No	No
Integration (NCI/DBF) coefficients	No	No	Yes	Yes

If the histogram is enabled, the window information and configuration linked lists must be stored in a different bank. Histogram data must not span more than one bank.

The TriCore™ subsystem should not attempt to access this memory during TC3Ax SPU run time. If a processor does attempt to access the memory while the TC3Ax SPU is using it the TC3Ax SPU will generate an error to the TriCore™ subsystem. This is to avoid stalling the processor execution for an excessive period of time.

The configuration memory is organised in 8 byte words. All register fields used to store configuration memory addresses must contain 8 byte aligned values. Any non-zero value written to bits[2:0] of these register fields will be ignored.

The configuration memory does not support byte or half-word (16 bit) write accesses.

## Signal Processing Unit 2 (SPU2)

## 20.4 Registers

Table 825 Register Overview - SPU\_2 (ascending Offset Address)

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
CLC	Clock Control	00000 <sub>H</sub>	U,SV	U,SV,E,P	See page 73	73
MODID	Module Identification Register	00004 <sub>H</sub>	U,SV	BE	Application Reset	74
STAT	Status and Reporting	00008 <sub>H</sub>	U,SV	U,SV,P	See page 75	75
STATCTRL	Status and Reporting Control	0000C <sub>H</sub>	U,SV	U,SV,P	See page 77	77
ID_CONF	Input DMA Configuration	00010 <sub>H</sub>	U,SV	U,SV,P	See page 77	77
ID_CONF2	Input DMA Configuration 2	00014 <sub>H</sub>	U,SV	U,SV,P	See page 78	78
ID_RM_CONF	Input DMA Configuration: Radar Memory	00018 <sub>H</sub>	U,SV	U,SV,P	See page 80	80
ID_RM_CONF2	Input DMA Configuration: Radar Memory	0001C <sub>H</sub>	U,SV	U,SV,P	See page 81	81
ID_RM_ILO	Inner Loop Address Offset	00020 <sub>H</sub>	U,SV	U,SV,P	See page 85	85
ID_RM_OLO	Outer Loop Address Offset	00024 <sub>H</sub>	U,SV	U,SV,P	See page 86	86
ID_RM_BLO	Bin Offset Address Configuration	00028 <sub>H</sub>	U,SV	U,SV,P	See page 86	86
ID_RM_IOLR	Inner and Outer Loop Repeat	0002C <sub>H</sub>	U,SV	U,SV,P	See page 87	87
ID_RM_BLR	Bin Loop Repeat	00030 <sub>H</sub>	U,SV	U,SV,P	See page 88	88
ID_RM_ACFG0	Spare Configuration Register	00034 <sub>H</sub>	U,SV	BE	See page 89	89
ID_RM_ACFG1	Spare Configuration Register	00038 <sub>H</sub>	U,SV	BE	See page 90	90
ID_BYPASS_CTRL	Bypass Control	0003C <sub>H</sub>	U,SV	U,SV,P	See page 91	91
PACTR	Partial-Acquisition Counter	00040 <sub>H</sub>	U,SV	U,SV,P	See page 92	92
DPASS_CONF	Double Pass Configuration	00044 <sub>H</sub>	U,SV	U,SV,P	See page 93	93
PCFG0	Reserved Configuration Register	00048 <sub>H</sub>	U,SV	BE	See page 95	95
PCFG1	Reserved Configuration Register	0004C <sub>H</sub>	U,SV	BE	See page 96	96
PCFG2	Reserved Configuration Register	00050 <sub>H</sub>	U,SV	BE	See page 96	96
PCFG3	Reserved Configuration Register	00054 <sub>H</sub>	U,SV	BE	See page 97	97
M0CTRL	MATH0 Control	00058 <sub>H</sub>	U,SV	U,SV,P	See page 98	98
TRTHRESH	Transient Removal Threshold	0005C <sub>H</sub>	U,SV	U,SV,P	See page 99	99
FILTCTRL	Filter control	00060 <sub>H</sub>	U,SV	U,SV,P	See page 100	100

## Signal Processing Unit 2 (SPU2)

Table 825 Register Overview - SPU\_2 (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
FILTCOEFc	Filter coefficient	00064 <sub>H</sub> + c*4	U,SV	U,SV,P	See page 101	101
DETCTRL	Detector control	00084 <sub>H</sub>	U,SV	U,SV,P	See page 102	102
DETBASE	Detector Base Address Configuration	00088 <sub>H</sub>	U,SV	U,SV,P	See page 104	104
DETCFG	Interference detector configuration	0008C <sub>H</sub>	U,SV	U,SV,P	See page 105	105
RPLCTRL	Control of the replacement function	00090 <sub>H</sub>	U,SV	U,SV,P	See page 105	105
BURSTb	Burst descriptor	00094 <sub>H</sub> + b*4	U,SV	U,SV,P	See page 107	107
SCRECIP	DC offset sample count reciprocal	000B4 <sub>H</sub>	U,SV	U,SV,P	See page 108	108
ZI_MASKn	Zero Insertion Mask	000B8 <sub>H</sub> + n*4	U,SV	U,SV,P	See page 109	109
BEx_LDR_CONF	Loader Configuration	000D8 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 110	110
BEx_LDR_CONF2	Loader Configuration Extended	000DC <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 111	111
BEx_Aj_ANTOFST	Antenna Offset	000E0 <sub>H</sub> + x*84 <sub>H</sub> +j* 4	U,SV	U,SV,P	See page 113	113
BEx_UNLDR_CO_NF	Unloader Configuration	000F0 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 114	114
BEx_UNLDR_CO_NF2	Unloader Configuration 2	000F4 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 115	115
BEx_UNLDR_CO_NF3	Unloader Configuration 3	000F8 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 116	116
BEx_UNLDR_ACFG	Spare Configuration Register	000FC <sub>H</sub> + x*84 <sub>H</sub>	U,SV	BE	See page 117	117
BEx_ODP_BASE	Output Data Processor Base Write Address	00100 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 118	118
BEx_ODP_CONF	Output Data Processor Configuration	00104 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 119	119
BEx_ODP_ILO	ODP Inner Loop Address Offset	00108 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 121	121
BEx_ODP_OLO	ODP Outer Loop Address Offset	0010C <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 122	122
BEx_ODP_BLO	ODP Bin Offset Address Configuration	00110 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 123	123

## Signal Processing Unit 2 (SPU2)

Table 825 Register Overview - SPU\_2 (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
BEx_ODP_IOLR	ODP Inner and Outer Loop Repeat	00114 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 124	124
BEx_ODP_BLR	ODP Bin Loop Repeat	00118 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 125	125
BEx_ND_BASEb	NCI DBF Base Address Configuration	0011C <sub>H</sub> + x*84 <sub>H</sub> +b*4	U,SV	U,SV,P	See page 126	126
BEx_NDCTRL	Dual Integration Unit Control	00124 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 127	127
BEx_NDCBASE	NCI and DBF Coefficient Base address	00128 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 130	130
BEx_PWRCTRL	Power Information Channel Control	0012C <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 131	131
BEx_CFARCTRL	CFAR Module Control	00130 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 132	132
BEx_CABASE	CFAR CA Engine Base Address	00134 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 135	135
BEx_GOSBASE	CFAR GOS Engine Base Address	00138 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 137	137
BEx_LCLMAXBAS E	Local Maximum Detection Base Address	0013C <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 138	138
BEx_LBLBASE	Label List Base Address	00140 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 139	139
BEx_LBLCALC	Label address calculation base address	00144 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 140	140
BEx_LBLX	Label X multiplier	00148 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 141	141
BEx_LBLY	Label Y multiplier	0014C <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 142	142
BEx_LBLZ	Label Z multiplier	00150 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 142	142
BEx_LBL	Label repeat values for X and Y directions	00154 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 143	143
BEx_SBCTRL	Sideband Control	00158 <sub>H</sub> + x*84 <sub>H</sub>	U,SV	U,SV,P	See page 144	144
BINm_REJ	Bin Rejection Mask	001E0 <sub>H</sub> + m*4	U,SV	U,SV,P	See page 145	145
BINREJCTRL	Bin Rejection Unit Control	002E0 <sub>H</sub>	U,SV	U,SV,P	See page 146	146
MAGAPPROX	Magnitude Approximation Constants	002E4 <sub>H</sub>	U,SV	U,SV,P	See page 148	148
SCALARADD	Scalar Addition Operand	002E8 <sub>H</sub>	U,SV	U,SV,P	See page 148	148

## Signal Processing Unit 2 (SPU2)

Table 825 Register Overview - SPU\_2 (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
SCALARMULT	Scalar Multiplication Operand	002EC <sub>H</sub>	U,SV	U,SV,P	See page 149	149
CFARCFG1	CFAR Configuration	002F0 <sub>H</sub>	U,SV	U,SV,P	See page 150	150
CFARCFG2	CFAR Configuration 2	002F4 <sub>H</sub>	U,SV	U,SV,P	See page 151	151
CFARCFG3	CFAR Configuration 3	002F8 <sub>H</sub>	U,SV	U,SV,P	See page 153	153
LCLMAX	Local Maximum Control	002FC <sub>H</sub>	U,SV	U,SV,P	See page 154	154
AUXCTRL	Auxiliary Control Information	00300 <sub>H</sub>	U,SV	U,SV,P	See page 156	156
ACFG0	Spare Configuration Register	00304 <sub>H</sub>	U,SV	BE	See page 157	157
ACFG1	Spare Configuration Register	00308 <sub>H</sub>	U,SV	BE	See page 158	158
REGCRC	Register CRC	0030C <sub>H</sub>	U,SV	U,SV,P	See page 158	158
CTRL	SPU Control	00310 <sub>H</sub>	U,SV	U,SV,P	See page 159	159
MDM0_CNT	Burst Count	00314 <sub>H</sub>	U,SV	BE	See page 161	161
MDM0_BURSTb	Burst descriptor	00318 <sub>H</sub> + b*4	U,SV	BE	See page 162	162
MDq_LBLCNT	Label List Counter	00358 <sub>H</sub> + q*54 <sub>H</sub>	U,SV	U,SV,P	See page 163	163
MDq_METADATA	Dataset Metadata	0035C <sub>H</sub> + q*54 <sub>H</sub>	U,SV	BE	See page 164	164
MDq_SPAREm	Reserved	00360 <sub>H</sub> + q*54 <sub>H</sub> +m *4	U,SV	BE	See page 166	166
IDLCNT	Idle Count	00400 <sub>H</sub>	U,SV	BE	See page 166	166
EXCNT	Execution Count	00404 <sub>H</sub>	U,SV	BE	See page 167	167
SMSTAT	Safety Mechanism Status	00408 <sub>H</sub>	U,SV	U,SV,P	See page 168	168
SMCTRL	Safety Mechanism Control Functions	0040C <sub>H</sub>	U,SV	U,SV,P	See page 169	169
MONITOR	SPU Monitor	00410 <sub>H</sub>	U,SV	BE	See page 172	172
SMUSER	Safety Mechanism Control Functions (User)	00414 <sub>H</sub>	U,SV	U,SV,P	See page 173	173
DATAd_CRC	Monitor CRC Register	00418 <sub>H</sub> + d*4	U,SV	BE	See page 175	175
CRC_DATA_CRC	CRC of Data Monitor CRC Registers	00570 <sub>H</sub>	U,SV	BE	See page 176	176
CTRLe_CRC	Monitor CRC Register	00590 <sub>H</sub> + e*4	U,SV	BE	See page 177	177
CRC_CTRL_CRC	CRC of Control Monitor CRC Registers	005F4 <sub>H</sub>	U,SV	BE	See page 177	177

## Signal Processing Unit 2 (SPU2)

**Table 825 Register Overview - SPU\_2 (ascending Offset Address) (cont'd)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
CRC_MASK0	Control CRC Mask Register	005F8 <sub>H</sub>	U,SV	U,SV,P	See page 178	178
CRC_MASK1	Spare Control CRC Mask Register	005FC <sub>H</sub>	U,SV	BE	See page 179	179
USRRTC	User OCDS Trace Control	007E0 <sub>H</sub>	U,SV	P,SV	See page 180	180
ACCEN0	Access Enable Register 0	007E4 <sub>H</sub>	U,SV	U,SV,SE	See page 181	181
ACCEN1	Access Enable Register 1	007E8 <sub>H</sub>	U,SV	U,SV,SE	See page 181	181
OCS	OCDS Control and Status	007EC <sub>H</sub>	U,SV	SV,OEN,P	See page 182	182
ODA	OCDS Debug Access Register	007F0 <sub>H</sub>	U,SV	SV,P	See page 184	184
KRST0	Kernel Reset Register 0	007F4 <sub>H</sub>	U,SV	U,SV,P	See page 185	185
KRST1	Kernel Reset Register 1	007F8 <sub>H</sub>	U,SV	U,SV,P	See page 186	186
KRSTCLR	Kernel Reset Clear	007FC <sub>H</sub>	U,SV	U,SV,P	See page 187	187

### 20.4.1 Register Description

#### Clock Control

<b>CLC</b> <b>Clock Control</b> <span style="float: right;">Reset Value: <a href="#">Table 827</a></span>															
<b>(00000<sub>H</sub>)</b>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES												<b>EDIS</b>	<b>FDIS</b>	<b>DISS</b>	<b>DISR</b>
										rw		rw	rh	rw	
r															

Field	Bits	Type	Description
<b>DISR</b>	0	rw	<b>Disable Request</b> 0 <sub>B</sub> <b>Enable</b> , Request that the SPU clock tree be switched on 1 <sub>B</sub> <b>Disable</b> , Request that the SPU clock tree be switched off
<b>DISS</b>	1	rh	<b>Disable Status</b> This bit will be set to 1 if the SPU kernel clock is disabled 0 <sub>B</sub> <b>Enable</b> , The SPU clock tree is switched on 1 <sub>B</sub> <b>Disabled</b> , The SPU clock tree is switched off

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>FDIS</b>	2	rw	<b>Freeze Disable</b> This bit controls the freeze function for this module. The freeze function is not implemented for the SPU so this bit will have no effect. This bit can be set to 0 <sub>B</sub> 0 <sub>B</sub> <b>Enable</b> , Module operates on uncorrected clock, with full modulation jitter. 1 <sub>B</sub> <b>Disabled</b> , Module Operates on corrected clock with reduced modulation jitter. No effect for SPU
<b>EDIS</b>	3	rw	<b>Sleep Mode Enable Control</b> Reserved. This bit currently has no effect on the SPU. It should be kept at 0 <sub>B</sub> for future compatibility 0 <sub>B</sub> <b>Disabled</b> , Sleep Mode is Off 1 <sub>B</sub> <b>Enable</b> , Sleep Mode is On (no effect)
<b>RES</b>	31:4	r	<b>Reserved</b>

**Table 826 Access Mode Restrictions of CLC sorted by descending priority**

Mode Name	Access Mode		Description
ENDINIT and Master enabled in ACCEN	r	RES	Write Accesses Permitted during Initialisation
	rh	DISS	
	rw	DISR, EDIS, FDIS	
Otherwise (default)	r	DISR, EDIS, FDIS, RES	Default Access Mode (Bus Error on Write)
	rh	DISS	

**Table 827 Reset Values of CLC**

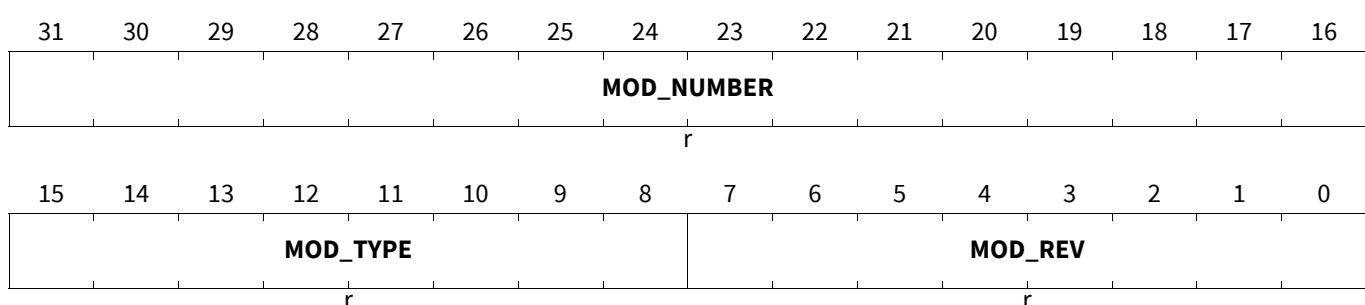
Reset Type	Reset Value	Note
Application Reset	0000 0003 <sub>H</sub>	Application Reset

### Module Identification Register

This register contains information intended to enable identification of the module and the version of the module. The Revision Number (MOD\_REV) of the SPU is 01<sub>H</sub> for TC39x, TC35x and TC33x and 02<sub>H</sub> for TC3Ax.

#### MODID

**Module Identification Register** (00004<sub>H</sub>) Application Reset Value: 00E4 C002<sub>H</sub>



## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>MOD_REV</b>	7:0	r	<b>Module Revision Number</b> This bit field defines the module revision number. The value of a module revision starts with 01H (first revision)
<b>MOD_TYPE</b>	15:8	r	<b>Module Type</b> Set to 0xC0 to indicate a 32 bit module
<b>MOD_NUMBE R</b>	31:16	r	<b>Module Number Value</b> Set to 0x00E4. This number is unique to the SPU.

**Table 828 Access Mode Restrictions of MODID sorted by descending priority**

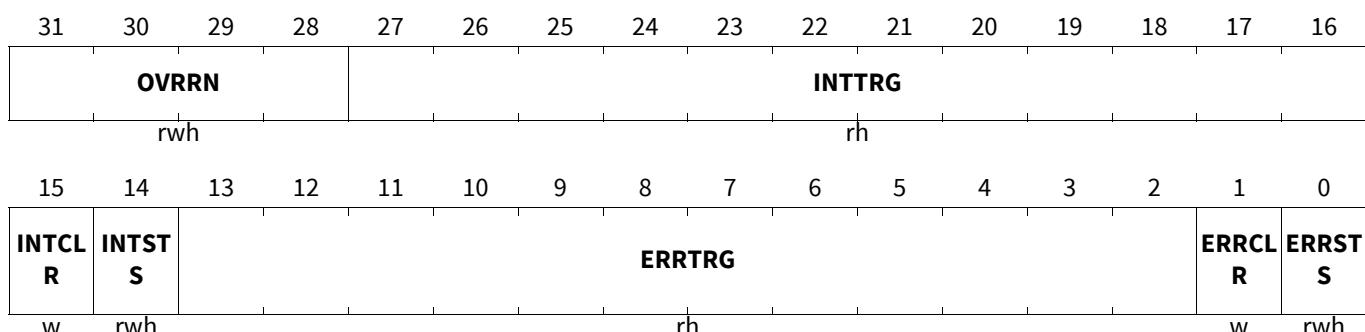
Mode Name	Access Mode		Description
Write Accesses	r	MOD_NUMBER, MOD_REV, MOD_TYPE	Any write access will cause bus error
Otherwise (default)	r	MOD_NUMBER, MOD_REV, MOD_TYPE	Default Access Mode (Bus Error on Write)

### Status and Reporting

This register allows the interrupt trigger masks to be set and the triggers causing interrupts to be read. See [Section 20.3.7.7, Interrupts](#) for bit allocation in the ERRMSK, INTMSK, ERRTRG and INTTRG fields

#### STAT

**Status and Reporting** **Reset Value:** [Table 830](#)



Field	Bits	Type	Description
<b>ERRSTS</b>	0	rwh	<b>Error Status</b> Set when an enabled error condition has triggered an interrupt
<b>ERRCLR</b>	1	w	<b>Error Clear</b> Set this bit to 1 while writing 0 to ERRSTS to clear the ERRSTS flag. Always reads as 0
<b>ERRTRG</b>	13:2	rh	<b>Interrupt Trigger</b> The bits in this bitfield correspond to the bits in ERRMSK. Bits set will indicate which condition or conditions have triggered an interrupt. This field will be cleared when ERRSTS is cleared

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>INTSTS</b>	14	rwh	<b>Interrupt Status</b> Set to 1 when an enabled interrupt condition (INTMSK) has triggered an interrupt
<b>INTCLR</b>	15	w	<b>Interrupt Clear</b> Set this bit to 1 while writing 0 to INTSTS to clear the Interrupt Status Flag. Always reads as 0
<b>INTTRG</b>	27:16	rh	<b>Interrupt Trigger</b> The bits in this bitfield correspond to the bits in INTMSK. Bits set will indicate which condition or conditions have triggered an interrupt. This field will be cleared when INTSTS is cleared
<b>OVRRN</b>	31:28	rwh	<b>Overrun</b> The maximum number of contiguous datasets skipped by the Input Data Manager due to overrun in the FFT processing. This field will continue to accumulate the maximum number of ramps skipped sequentially until it is explicitly cleared by writing 0. Any value written during configuration will be overwritten at this point.

**Table 829 Access Mode Restrictions of STAT sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rh	ERRTRG, INTTRG	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rwh	ERRSTS, INTSTS, OVRRN	
	w	ERRCLR, INTCLR	
Otherwise (default)	rX	ERRCLR, INTCLR	Default Access Mode (Bus Error on Write)
	rh	ERRSTS, ERRTRG, INTSTS, INTTRG, OVRRN	

**Table 830 Reset Values of STAT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit 2 (SPU2)

### Status and Reporting Control

#### STATCTRL

**Status and Reporting Control** **(0000C<sub>H</sub>)** **Reset Value:** [Table 832](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>RES28</b>				<b>INTMSK</b>											
r															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>RES14</b>		<b>ERRMSK</b>								<b>RES0</b>				r	
r															

Field	Bits	Type	Description
<b>RES0</b>	1:0	r	<b>Reserved</b>
<b>ERRMSK</b>	13:2	rw	<b>Error Mask</b> Mask field to enable interrupt on particular error conditions
<b>RES14</b>	15:14	r	<b>Reserved</b>
<b>INTMSK</b>	27:16	rw	<b>Interrupt Mask</b> Mask Field to Enable/Disable Service Interrupt on particular events or conditions
<b>RES28</b>	31:28	r	<b>Reserved</b>

**Table 831 Access Mode Restrictions of STATCTRL sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ERRMSK, INTMSK	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ERRMSK, INTMSK	Default Access Mode (Bus Error on Write)

**Table 832 Reset Values of STATCTRL**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Input DMA Configuration

This register configures the base operating mode for the Input DMA and also provides the configuration parameters specific to accepting data from the RIF(s). The ANT, SMPLCNT, RAMPS and FORMAT fields are only used if the data source in the SRC field is specified as one or more RIFs. These fields will be ignored if the SRC field specifies that data is to be read from Radar Memory

## Signal Processing Unit 2 (SPU2)

### ID\_CONF

#### Input DMA Configuration

(00010<sub>H</sub>)

Reset Value: [Table 834](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								RAMPS							
r															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								SMPLCNT							
r															rw

Field	Bits	Type	Description
SMPLCNT	10:0	rw	<b>Sample Count</b> Number of samples per ramp (bin loop count). This field contains the number of data samples per ramp minus one. i.e. for a 512 point data set programme a value of 511.
RAMPS	26:16	rw	<b>Ramps per Measurement Cycle</b> The number of ramps expected to be input for this measurement cycle. The IDM will process this number of ramps and write the data to the buffer memory. Excess data will be silently ignored unless the SPU is retriggered. The number of ramps expected is the contents of this bitfield plus one. i.e. 0 equates to 1 ramp and 2047 equates to 2048 ramps.
0	15:11, 31:27	r	<b>Reserved</b>

**Table 833 Access Mode Restrictions of ID\_CONF sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	RAMPS, SMPLCNT	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	RAMPS, SMPLCNT	Default Access Mode (Bus Error on Write)

**Table 834 Reset Values of ID\_CONF**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Input DMA Configuration 2

The Input DMA can bypass data from RIF1 if the incoming data for each ramp exceeds the capacity of the buffer memory. This register configures the bypass mode and also allows the bypassed data to be reloaded into the buffer memory by the Input DMA. Note that, when reloading bypassed data, the data source is still considered to be a RIF for configuration purposes.

## Signal Processing Unit 2 (SPU2)

## ID\_CONF2

## Input DMA Configuration 2

(00014<sub>H</sub>)Reset Value: [Table 836](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>FORM</b>	<b>SIGNE</b>							<b>0</b>							
<small>rw</small>	<small>rw</small>							<small>r</small>							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				<b>0</b>				<b>0</b>		<b>SRCRI F1</b>	<b>SRCRI F0</b>	<b>ANT</b>	<b>SRC</b>		
				<small>r</small>				<small>r</small>		<small>rw</small>	<small>rw</small>	<small>rw</small>	<small>rw</small>		<small>rw</small>

Field	Bits	Type	Description
<b>SRC</b>	1:0	rw	<p><b>Data Source</b></p> <p>The bitfield controls the source of the data to be processed</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> <b>RIF</b>, Data is pushed from the RIF enabled by the SRCRIF field</li> <li>01<sub>B</sub> <b>BPRLD</b>, Bypassed Data is reloaded</li> <li>Bypassed RIF data is reloaded. The start address for the data is stored in the BYPASS_CTRL.BPADDR bitfield. The SRCRIF, ANT and FORMAT bitfields must be set to appropriate values for the bypassed data</li> <li>10<sub>B</sub> <b>Reserved</b>, Reserved Value</li> <li>11<sub>B</sub> <b>RM</b>, Data will be read from Radar Memory</li> <li>Data will be read from Radar Memory. The ID_RM_* registers are used to configure this type of operation</li> </ul>
<b>ANT</b>	3:2	rw	<p><b>Number of Antennae</b></p> <p>The number of antennae connected to each RIF instance to be used by this SPU</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> <b>ONE</b>, One Antenna is connected</li> <li>01<sub>B</sub> <b>TWO</b>, Two Antennae are connected</li> <li>10<sub>B</sub> <b>THREE</b>, Three Antennae are connected</li> <li>11<sub>B</sub> <b>FOUR</b>, Four Antennae are connected</li> </ul>
<b>SRCRIFm (m=0-1)</b>	m+4	rw	<p><b>Source RIF Enabled</b></p> <p>Use RIF(m) as source for input data.</p>
<b>SIGNED</b>	30	rw	<p><b>Signed or Unsigned Data</b></p> <p>Defines whether the RIF input data will be treated as unsigned values or two's complement data.</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> <b>UNSIGNED</b>, Unsigned Data</li> <li>The Input Data from the RIF is unsigned. This allows 16 bits of resolution. i.e. the input data range is 0 to 2<sup>16</sup>-1</li> <li>1<sub>B</sub> <b>SIGNED</b>, Signed Data</li> <li>The Input Data from the RIF is in 2's complement format</li> </ul>

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>FORMAT</b>	31	rw	<b>RIF Data Format</b> RIF data is always 16 bit. This bitfield selects whether the data is real or complex. 0 <sub>B</sub> <b>REAL</b> , Input Data is a Real Number 1 <sub>B</sub> <b>COMPLEX</b> , Input Data has Real and Imaginary Components
<b>0</b>	7:6, 29:8	r	<b>Reserved</b>

**Table 835 Access Mode Restrictions of ID\_CONF2 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ANT, FORMAT, SIGNED, SRC, SRCRIFm (m=0-1)	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ANT, FORMAT, SIGNED, SRC, SRCRIFm (m=0-1)	Default Access Mode (Bus Error on Write)

**Table 836 Reset Values of ID\_CONF2**

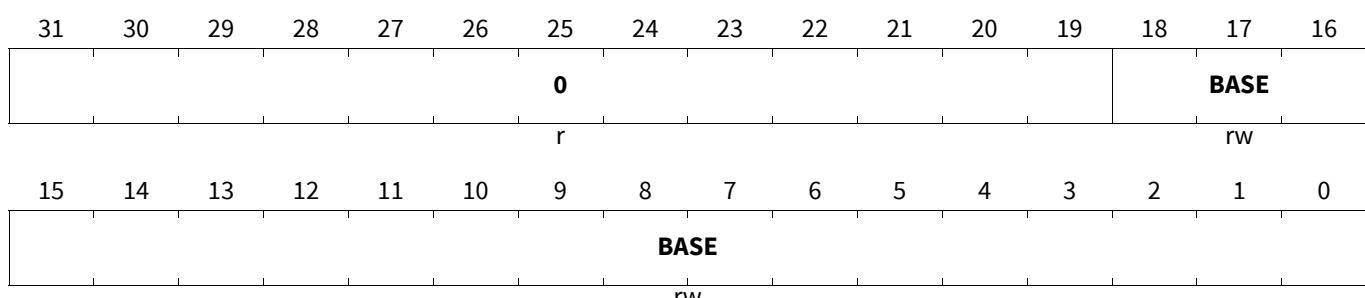
Reset Type	Reset Value	Note
Application Reset	4000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	4000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Input DMA Configuration: Radar Memory

This register contains options used to configure the Input DMA for reading data from Radar Memory. Register field descriptions should be read in conjunction with [Section 20.3.1.2.1, Principles of Operation \(Radar Memory\)](#) where usage and permissible combinations are more fully defined

#### ID\_RM\_CONF

**Input DMA Configuration: Radar Memory (00018<sub>H</sub>) Reset Value: Table 838**



## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> Base Address of the data to be read from Radar Memory. This is a 32 byte (256 bit) aligned address relative to the start address of the Radar Memory. Five LSBs need to be added to convert to a system address. This field is sized to address a 16 MiB Radar Memory. Accessing locations outside the range of the memory on the device will result in an error being generated by the Radar Memory module.
<b>0</b>	31:19	r	<b>Reserved</b>

**Table 837 Access Mode Restrictions of ID\_RM\_CONF sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BASE	Default Access Mode (Bus Error on Write)

**Table 838 Reset Values of ID\_RM\_CONF**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Input DMA Configuration: Radar Memory

This register contains options used to configure the Input DMA for reading data from Radar Memory. Register field descriptions should be read in conjunction with [Section 20.3.1.2.1, Principles of Operation \(Radar Memory\)](#) where usage and permissible combinations are more fully defined

#### ID\_RM\_CONF2

**Input DMA Configuration: Radar Memory (0001C<sub>H</sub>) Reset Value: Table 840**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								0							
									r						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>0</b>	<b>IMODE</b>	<b>0</b>	<b>PM</b>	<b>0</b>	<b>BLOCKS</b>			<b>AM</b>		<b>TRNS PS</b>		<b>FORMAT</b>			
r	rw	r	rw	r				rw		rw		rw			

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
FORMAT	3:0	rw	<p><b>Input Data Format</b></p> <p>The format of the Input Data to be read from the Radar Memory. Note that the REAL16BIT format must not be used if the TRNSPS bit is also set.</p> <ul style="list-style-type: none"> <li>0<sub>H</sub> <b>CMPLX16BIT</b>, 16 Bit Precision Complex Data</li> <li>1<sub>H</sub> <b>CMPLX32BIT</b>, 32 Bit Precision Complex Data</li> <li>2<sub>H</sub> <b>PWR32BIT</b>, 32 Bit Precision Power Data</li> <li>3<sub>H</sub> <b>COMPRESS</b>, Compressed Data Format</li> <li>4<sub>H</sub> <b>REAL16BIT</b>, 16 Bit Precision Real Data The format cannot be used if ID_RM_CONF.TRNSPS=1</li> <li>5<sub>H</sub> <b>REAL32BIT</b>, 32 Bit Precision Real Data</li> <li>6<sub>H</sub> <b>REAL16FP</b>, Half Precision Floating Point, real data The format cannot be used if ID_RM_CONF.TRNSPS=1</li> <li>7<sub>H</sub> <b>CMPLX16FP</b>, Half Precision Floating Point, complex data</li> <li>8<sub>H</sub> <b>PWR16FP</b>, Half Precision Floating Point, power data</li> <li>9<sub>H</sub> <b>RES9</b>,</li> <li>...</li> <li>F<sub>H</sub> <b>RES15</b>,</li> </ul>
TRNSPS	4	rw	<p><b>Transpose Addressing</b></p> <p>This bit should be set when the bins or samples used to construct an FFT input set in the Buffer memory do not occupy contiguous addresses in the Radar Memory. If set, it switches the operating mode of the Input DMA so that each of the bins in the 256 bit word read from radar Memory are treated as bins of different input FFT datasets when being written to buffer memory</p> <ul style="list-style-type: none"> <li>0<sub>B</sub> <b>LIN</b>, Linear Mode Bins stored in contiguous addresses of Radar Memory will be loaded directly into Buffer Memory as an input for an FFT dataset.</li> <li>1<sub>B</sub> <b>TRN</b>, Transpose Bins stored in contiguous addresses of Radar Memory will be treated as belonging to different FFT input datasets</li> </ul>

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>AM</b>	7:5	rw	<p><b>Antenna Mapping</b></p> <p>Several processing elements of the SPU need to identify each FFT dataset with a particular antenna. This field, in conjunction with the PM field, defines how this should be done. This field applies when PM is set to 0<sub>B</sub>. Setting PM to 1<sub>B</sub> is equivalent to setting this field to IDX.</p> <p>000<sub>B</sub> <b>OFF</b>, Default            Antenna ID passed to processing chain is permanently 0<sub>D</sub></p> <p>001<sub>B</sub> <b>IDX</b>, Index Mode            Antenna ID is derived from the dataset index</p> <p>010<sub>B</sub> <b>ILR</b>, Inner Loop Repeat            Antenna ID is derived from the Inner Loop Repeat Counter Value</p> <p>011<sub>B</sub> <b>OLR</b>, Outer Loop Repeat            Antenna ID is derived from the Outer Loop Repeat Counter Value</p> <p>100<sub>B</sub> <b>BLR</b>, Bin Loop Repeat            Antenna ID is derived from the Bin Loop Repeat Counter Value</p> <p>101<sub>B</sub> <b>RES5</b>, Reserved</p> <p>...</p> <p>111<sub>B</sub> <b>RES7</b>, Reserved</p>
<b>BLOCKS</b>	10:8	rw	<p><b>Number of Datablocks</b></p> <p>In Integration mode, the number of datablocks to be simultaneously constructed in the buffer memory. Normally when reading across antenna, most of the data read is discarded. If this field is non-zero, the Input DMA Engine will attempt to build multiple datablocks in memory using the additional data. The value of this field is then used to control the maximum number of datablocks that will fit into the physical limit of the buffer memory. The number of datablocks is the field value plus one. The maximum permissible value for this field is 3 for 64 bit data (32 bit precision complex) and 7 for 32 bit data (16 bit precision complex or 32 bit power)</p>

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
PM	12	rw	<p><b>Processing Mode</b>  This bit is set when the Input DMA is required to build datablocks comprising one FFT from each attached antennae. This would be necessary when it is intended to use the coherent or non-coherent integration functions of the MATH2 unit.</p> <p>0<sub>B</sub> <b>DM</b>, Default Mode  This is the optimal mode for in-place FFT calculation. All the datapoints read from memory can be directly used to construct a single datablock in the buffer memory. This means that the 32<sub>D</sub> or 64<sub>D</sub> bit data read in each 256<sub>D</sub> bit word will be used by incrementing either the bin loop or inner loop</p> <p>1<sub>B</sub> <b>IM</b>, Integration Mode  In Integration Mode, it is assumed that the buffer memory must contain one FFT input dataset for each antenna. Setting this bit forces the buffer memory to be switched only after the completion of a complete pass through the inner loop. If this bit is set, then the Input DMA Engine is allowed to construct multiple datablocks in memory if this is needed to maximise utilisation of the Radar Memory bandwidth. This function is controlled by the ID_RM_CONF.BLOCKS bitfield.</p>
IMODE	14	rw	<p><b>Incremental Mode</b>  Incremental Processing Mode. If enabled, the IDM will pause before first loading the Buffer RAM and after each Buffer RAM switch until a pulse has been detected on the Incremental Trigger input from another SPU. The address generation parameters used in the ODM of the other SPU must be an exact match to the IDM address generation parameters in this SPU. The SPU will keep track of the number of outstanding incremental trigger events if the other SPU is processing data faster than the SPU.</p> <p>0<sub>B</sub> <b>OFF</b>, Incremental Mode is Disabled  1<sub>B</sub> <b>ON</b>, Incremental Mode is Enabled</p>
0	11, 13, 31:15	r	<b>Reserved</b>

Table 839 Access Mode Restrictions of ID\_RM\_CONF2 sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	AM, BLOCKS, FORMAT, IMODE, PM, TRNSPS	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	AM, BLOCKS, FORMAT, IMODE, PM, TRNSPS	Default Access Mode (Bus Error on Write)

## Signal Processing Unit 2 (SPU2)

**Table 840 Reset Values of ID\_RM\_CONF2**

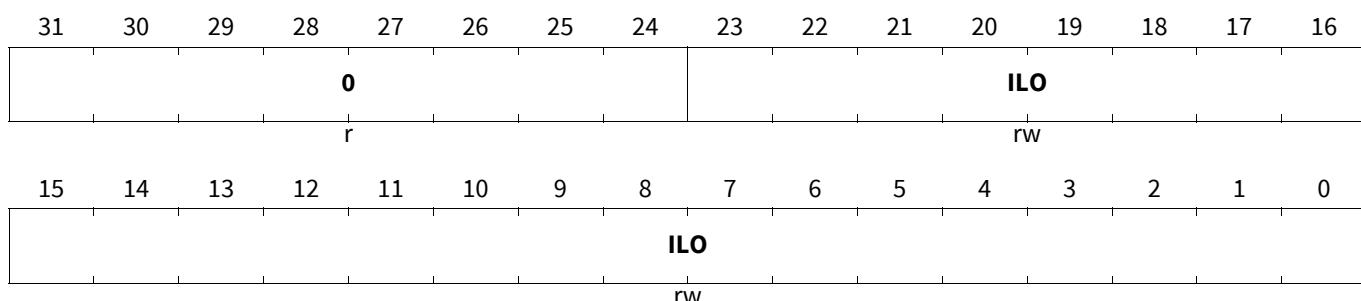
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Inner Loop Address Offset

Options used to configure the Input DMA for reading data from Radar Memory. Register field descriptions should be read in conjunction with [Section 20.3.1.2.1, Principles of Operation \(Radar Memory\)](#) where usage and permissible combinations are more fully defined

#### ID\_RM\_ILO

**Inner Loop Address Offset** **(00020<sub>H</sub>)** **Reset Value: Table 842**



Field	Bits	Type	Description
ILO	23:0	rw	<b>Inner Loop Offset</b> The Inner Loop Offset is a byte address. The smallest permissible value is the size of a single data element (e.g. a value of eight for a 32 bit precision complex data word)
0	31:24	r	<b>Reserved</b>

**Table 841 Access Mode Restrictions of ID\_RM\_ILO sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ILO	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ILO	Default Access Mode (Bus Error on Write)

**Table 842 Reset Values of ID\_RM\_ILO**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit 2 (SPU2)

### Outer Loop Address Offset

Options used to configure the Input DMA for reading data from Radar Memory. Register field descriptions should be read in conjunction with [Section 20.3.1.2.1, Principles of Operation \(Radar Memory\)](#) where usage and permissible combinations are more fully defined

#### ID\_RM\_OLO

**Outer Loop Address Offset** **(00024<sub>H</sub>)** **Reset Value: Table 844**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								OLO							
r															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OLO								rw							

Field	Bits	Type	Description
OLO	23:0	rw	<b>Outer Loop Offset</b> The Outer Loop Offset is a byte address. The smallest permissible value is the size of a single data element (e.g. a value of eight for a 32 bit precision complex data word)
0	31:24	r	<b>Reserved</b>

**Table 843 Access Mode Restrictions of ID\_RM\_OLO sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	OLO	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	OLO	Default Access Mode (Bus Error on Write)

**Table 844 Reset Values of ID\_RM\_OLO**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Bin Offset Address Configuration

Options used to configure the Input DMA for reading data from Radar Memory. Register field descriptions should be read in conjunction with [Section 20.3.1.2.1, Principles of Operation \(Radar Memory\)](#) where usage and permissible combinations are more fully defined

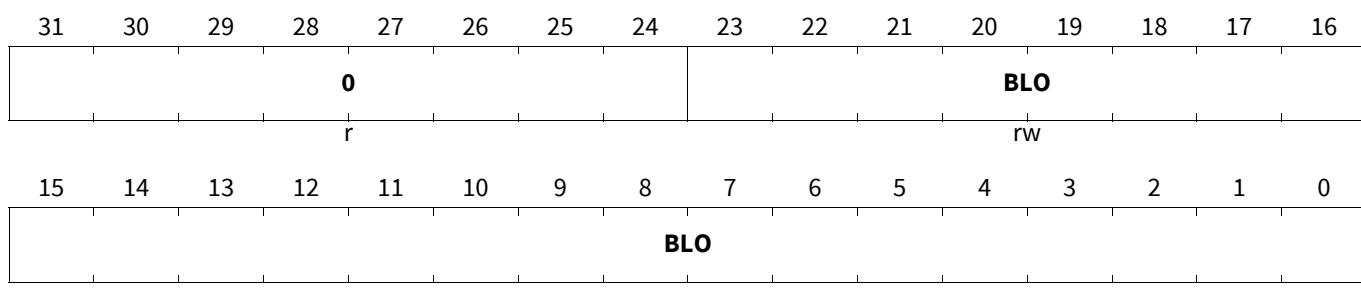
## Signal Processing Unit 2 (SPU2)

### ID\_RM\_BLO

#### Bin Offset Address Configuration

(00028<sub>H</sub>)

Reset Value: [Table 846](#)



Field	Bits	Type	Description
BLO	23:0	rw	<b>Bin Loop Offset</b> The Bin Loop Offset is a byte address. The smallest permissible value is the size of a single data element (e.g. a value of eight for a 32 bit precision complex data word)
0	31:24	r	<b>Reserved</b>

**Table 845 Access Mode Restrictions of ID\_RM\_BLO sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BLO	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BLO	Default Access Mode (Bus Error on Write)

**Table 846 Reset Values of ID\_RM\_BLO**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Inner and Outer Loop Repeat

Options used to configure the Input DMA for reading data from Radar Memory. The fields in this register define the number of Iterations for the Inner and Outer Addressing Loops. Register field descriptions should be read in conjunction with [Section 20.3.1.2.1, Principles of Operation \(Radar Memory\)](#) where usage and permissible combinations are more fully defined

## Signal Processing Unit 2 (SPU2)

## ID\_RM\_IOLR

Inner and Outer Loop Repeat

(0002C<sub>H</sub>)Reset Value: [Table 848](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			0												
OLR															
r															rw
			0												
ILR															
r															rw

Field	Bits	Type	Description
ILR	12:0	rw	<b>Inner Loop Repeat</b> Repetition Count for the Inner Address Loop. The loop will repeat at least once. The overall number of repetitions will be the value of this bitfield + 1.
OLR	28:16	rw	<b>Outer Loop Repeat</b> Repetition Count for the Outer Address Loop. The loop will repeat at least once. The overall number of repetitions will be the value of this bitfield + 1.
0	15:13, 31:29	r	<b>Reserved</b>

**Table 847 Access Mode Restrictions of ID\_RM\_IOLR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ILR, OLR	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ILR, OLR	Default Access Mode (Bus Error on Write)

**Table 848 Reset Values of ID\_RM\_IOLR**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

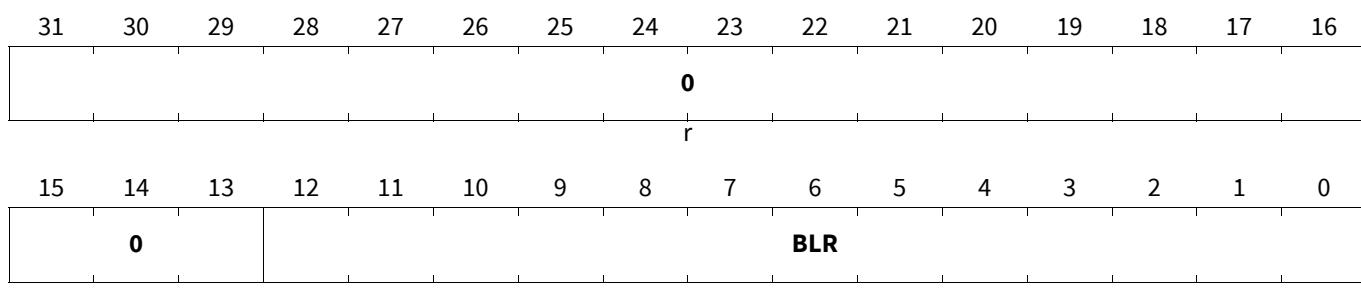
**Bin Loop Repeat**

Options used to configure the Input DMA for reading data from Radar Memory. The field in this register defines the number of Iterations for the Bin Addressing Loop. Register field descriptions should be read in conjunction with [Section 20.3.1.2.1, Principles of Operation \(Radar Memory\)](#) where usage and permissible combinations are more fully defined

## Signal Processing Unit 2 (SPU2)

### ID\_RM\_BLR

#### Bin Loop Repeat

(00030<sub>H</sub>)Reset Value: [Table 850](#)

Field	Bits	Type	Description
BLR	12:0	rw	<b>Bin Loop Repeat</b> Repetition Count for the Bin Address Loop. The loop will repeat at least once. The overall number of repetitions will be the value of this bitfield + 1.
0	31:13	r	<b>Reserved</b>

**Table 849 Access Mode Restrictions of ID\_RM\_BLR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BLR	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BLR	Default Access Mode (Bus Error on Write)

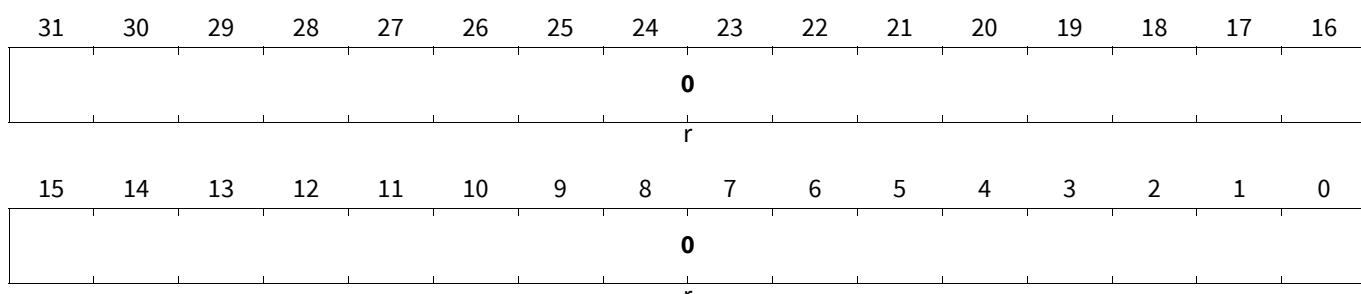
**Table 850 Reset Values of ID\_RM\_BLR**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Spare Configuration Register

#### ID\_RM\_ACFG0

#### Spare Configuration Register

(00034<sub>H</sub>)Reset Value: [Table 852](#)

**Signal Processing Unit 2 (SPU2)**

Field	Bits	Type	Description
0	31:0	r	<b>Reserved</b>

**Table 851** Access Mode Restrictions of **ID\_RM\_ACFG0** sorted by descending priority

<b>Mode Name</b>	<b>Access Mode</b>		<b>Description</b>
Write Accesses	-	See bit field definitions above	Any write access will cause bus error
Otherwise (default)	-	See bit field definitions above	Default Access Mode (Bus Error on Write)

**Table 852 Reset Values of ID\_RM\_ACFG0**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

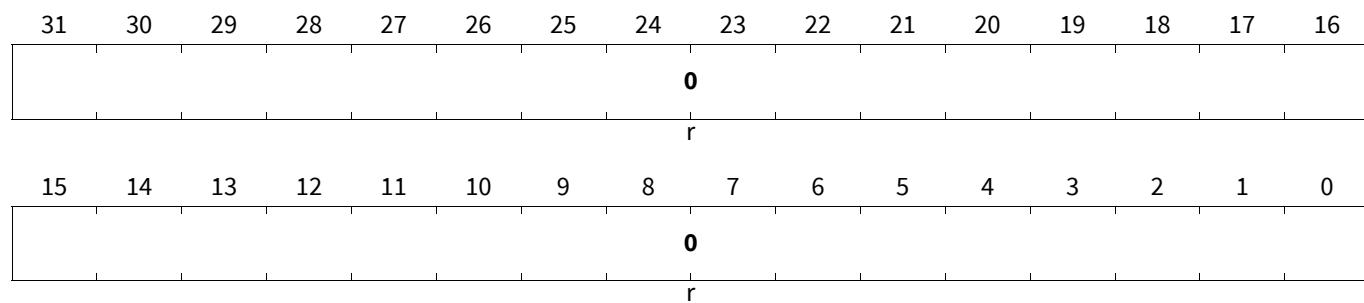
## Spare Configuration Register

ID\_RM\_ACFG1

## Spare Configuration Register

(00038<sub>H</sub>)

## **Reset Value: Table 854**



<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>0</b>	31:0	r	<b>Reserved</b>

**Table 853 Access Mode Restrictions of ID\_RM\_ACFG1 sorted by descending priority**

<b>Mode Name</b>	<b>Access Mode</b>		<b>Description</b>
Write Accesses	-	See bit field definitions above	Any write access will cause bus error
Otherwise (default)	-	See bit field definitions above	Default Access Mode (Bus Error on Write)

## Signal Processing Unit 2 (SPU2)

**Table 854 Reset Values of ID\_RM\_ACFG1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Bypass Control

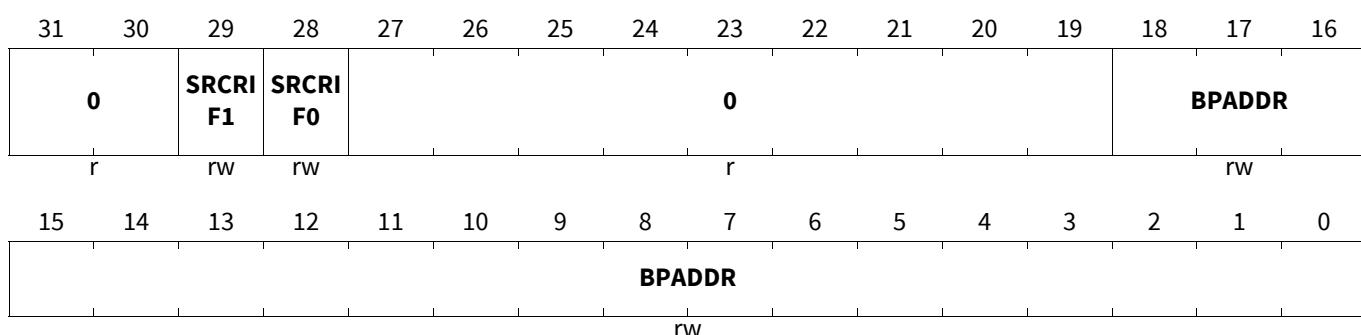
This register configures the bypass mode and also allows the bypassed data to be reloaded into the buffer memory by the Input DMA. Note that, when reloading bypassed data, the data source is still considered to be one or more RIFs for configuration purposes so the options controlling RIF data input in the ID\_CONF and ID\_CONF2 registers must be set to match the settings used to bypass the data to allow correct decoding of the data.

#### ID\_BYPASS\_CTRL

##### Bypass Control

(0003C<sub>H</sub>)

Reset Value: [Table 856](#)



Field	Bits	Type	Description
BPADDR	18:0	rw	<b>Bypass Address</b> This is the base address to be used when reading or writing bypass data. It is a 256 bit (32 byte) aligned address and needs to have five LSBs added to convert to a system address. It allows for a maximum addressable range of 16 MiB. Bypass data is written as a raw data stream as generated by the Radar Interface. No processing is performed by the SPU until the data is read back from the Radar Memory. This field is sized to address a 16 MiB Radar Memory. Accessing locations outside the range of the memory on the device will result in an error being generated by the Radar Memory module.
SRCRIFm (m=0-1)	m+28	rw	<b>Source RIF Enabled</b> Use RIF(m) as source for Bypass Data. At least one of these bits must be set to enable Bypass.
0	27:19, 31:30	r	<b>Reserved</b>

**Signal Processing Unit 2 (SPU2)**

**Table 855 Access Mode Restrictions of ID\_BYPASS\_CTRL sorted by descending priority**

<b>Mode Name</b>	<b>Access Mode</b>		<b>Description</b>
Master enabled in ACCEN	rw	BPADDR, SRCRIFm (m=0-1)	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BPADDR, SRCRIFm (m=0-1)	Default Access Mode (Bus Error on Write)

**Table 856** Reset Values of ID\_BYPASS\_CTRL

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Partial-Acquisition Counter

This counter is used to monitor and control partial-acquisition. If enabled, it will monitor the input data stream from the RIF and increment at the end of each ramp as determined by the sample count and number of antennae captured from the IDM configuration when the counter was last reset. As the counter increments at the end of the ramp, the first ramp of the measurement cycle will be considered as ramp 0.

Note: LDMST should be used only with bit mask enabled for the 'rh' bits in this register. Hardware updates occurring between the read and write phases of an RMW may be overwritten.

PACTR

## Partial-Acquisition Counter

(00040<sub>H</sub>)

### **Reset Value: Table 858**

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

0 COUNT

r rh

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

ATTN ERR TRIG LIMIT EN RST

rw rw rw rw rwh w

Field	Bits	Type	Description
RST	0	w	<b>Counter Reset</b> Write 1 to reset the Counter and Update the Control Information. Always reads as 0
EN	1	rwh	<b>Counter Enable</b> This field is used to enable the counter. It will only be updated if the counter is reset on the same write 0 <sub>B</sub> <b>DISABLE</b> , Disable the Counter 1 <sub>B</sub> <b>ENABLE</b> , Enable the Counter
LIMIT	12:2	rw	<b>Preacquisition Counter Limit</b> Limit value for the counter. For purposes of comparison, the first ramp of a measurement cycle is considered to be ramp 0

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
TRIG	13	rw	<b>Trigger on Limit</b> If this bit is set, then the SPU will start acquiring data when the counter reaches the value in the LIMIT field. This is considered as an internal trigger event and the CTRL.MODE bitfield should be set for internal trigger.
ERR	14	rw	<b>Error Interrupt on Limit</b> If this bit is set an error interrupt will be triggered if the counter reaches the value in the LIMIT field
ATTN	15	rw	<b>Attention Request Interrupt onLimit</b> If this bit is set an attention request interrupt will be triggered if the counter reaches the value in the LIMIT field
COUNT	26:16	rh	<b>Counter Value</b> The current value of the Partial Acquisition Counter
0	31:27	r	<b>Reserved</b> Always returns 0 when read

**Table 857 Access Mode Restrictions of PACTR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rh	COUNT	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	ATTN, ERR, LIMIT, TRIG	
	w	RST	
Master enabled in ACCEN and write 1 to RST	rwh	EN	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ATTN, ERR, LIMIT, TRIG	Default Access Mode (Bus Error on Write)
	rX	RST	
	rh	COUNT, EN	

**Table 858 Reset Values of PACTR**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

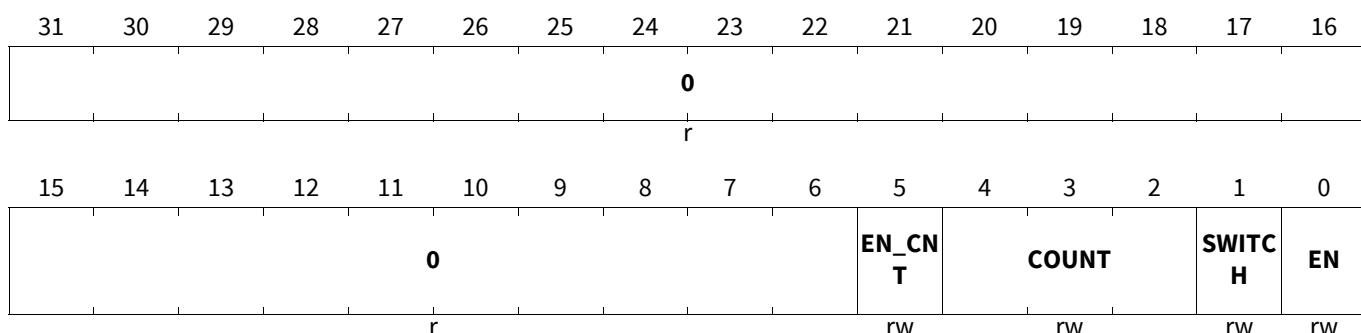
### Double Pass Configuration

Double Pass Mode allows the configuration of the MATH1 unit to be changed during processing of ADC data. The selected configuration can be alternated on a ramp-by ramp basis or both programmed configurations can be applied to all ramps.

## Signal Processing Unit 2 (SPU2)

### DPASS\_CONF

#### Double Pass Configuration

(00044<sub>H</sub>)Reset Value: [Table 860](#)

Field	Bits	Type	Description
<b>EN</b>	0	rw	<b>Enable</b> The SPU allows all processing operations to be run twice per data block or different processing parameters to be used on sequences of data blocks. This mode is enabled using the DPASS_CONF.EN bitfield. This results in two sets of results being written to Radar Memory
<b>SWITCH</b>	1	rw	<b>Buffer Memory Switch</b> If set to 0 <sub>B</sub> , both configurations of the MATH1 and MATH2 units will be applied to all data blocks and two full sets of output data will be written to the Radar Memory. If set to 1 <sub>B</sub> , the two configurations for MATH1 and MATH2 will be applied to alternate data blocks. Each set of output data written to Radar memory will contain half the data.
<b>COUNT</b>	4:2	rw	<b>Switch Count</b> If DPASS_CONF.EN_CNT is set, this field determines the number of data blocks to be processed before the alternate sets of window parameters are switched. The number of data blocks processed by the Loader between parameter switches is COUNT+1. i.e. if COUNT=0 then the window parameters will be switched for every data block processed, if COUNT=7, then the parameters will be switched after every eight data blocks processed.
<b>EN_CNT</b>	5	rw	<b>Enable Count</b> This bit enables switching the two sets of window parameters based on the DPASS_CONF.COUNT bitfield rather than the "Double Pass" settings. When set, the window parameters are switched every time a counter counting the number of data blocks processed reached the terminal count stored in the COUNT field + 1. The DPASS_CONF.EN and DPASS_CONF.SWITCH settings have no effect on the window parameters when this bit is set.
<b>0</b>	31:6	r	<b>Reserved</b>

## Signal Processing Unit 2 (SPU2)

**Table 859 Access Mode Restrictions of DPASS\_CONF sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	COUNT, EN, EN_CNT, SWITCH	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	COUNT, EN, EN_CNT, SWITCH	Default Access Mode (Bus Error on Write)

**Table 860 Reset Values of DPASS\_CONF**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Reserved Configuration Register

#### PCFG0

**Reserved Configuration Register** (00048<sub>H</sub>) Reset Value: Table 862

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								0							
									r						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								0							
									r						

Field	Bits	Type	Description
0	31:0	r	RESERVED

**Table 861 Access Mode Restrictions of PCFG0 sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	-	See bit field definitions above	Any write access will cause bus error
Otherwise (default)	-	See bit field definitions above	Default Access Mode (Bus Error on Write)

**Table 862 Reset Values of PCFG0**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit 2 (SPU2)

### Reserved Configuration Register

#### PCFG1

**Reserved Configuration Register (0004C<sub>H</sub>)** **Reset Value: Table 864**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															

Field	Bits	Type	Description
0	31:0	r	RESERVED

**Table 863 Access Mode Restrictions of PCFG1 sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	-	See bit field definitions above	Any write access will cause bus error
Otherwise (default)	-	See bit field definitions above	Default Access Mode (Bus Error on Write)

**Table 864 Reset Values of PCFG1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Reserved Configuration Register

#### PCFG2

**Reserved Configuration Register (00050<sub>H</sub>)** **Reset Value: Table 866**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															

Field	Bits	Type	Description
0	31:0	r	RESERVED

## Signal Processing Unit 2 (SPU2)

**Table 865 Access Mode Restrictions of PCFG2 sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	-	See bit field definitions above	Any write access will cause bus error
Otherwise (default)	-	See bit field definitions above	Default Access Mode (Bus Error on Write)

**Table 866 Reset Values of PCFG2**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Reserved Configuration Register

**PCFG3**

Reserved Configuration Register (00054 <sub>H</sub> )																Reset Value: <a href="#">Table 868</a>			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	0			
0																r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	0			
0																r			

Field	Bits	Type	Description
0	31:0	r	RESERVED

**Table 867 Access Mode Restrictions of PCFG3 sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	-	See bit field definitions above	Any write access will cause bus error
Otherwise (default)	-	See bit field definitions above	Default Access Mode (Bus Error on Write)

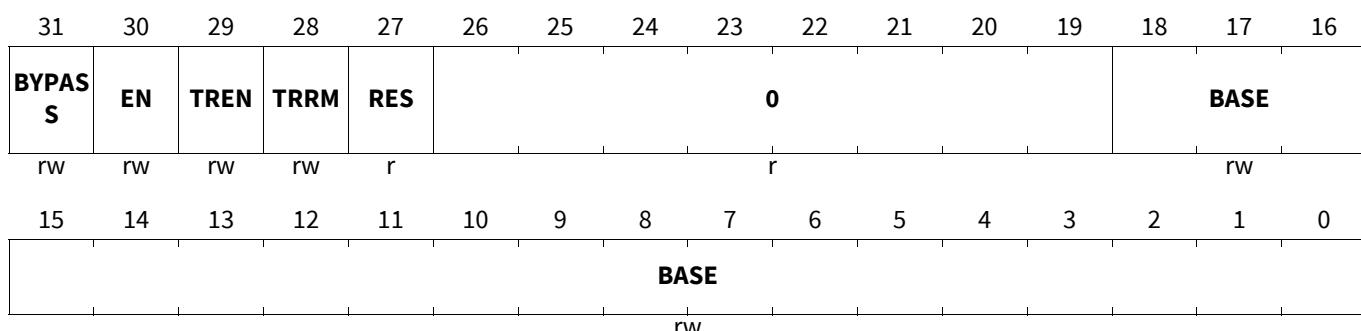
**Table 868 Reset Values of PCFG3**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit 2 (SPU2)

**MATH0 Control**

MATH0 control parameters

**M0CTRL****MATH0 Control**(000058<sub>H</sub>)Reset Value: [Table 870](#)

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>MATH0 Output Base Address</b> Base Address of where data shall be written to Radar Memory. This is a 32 byte (256 bit) aligned address relative to the start address of the Radar Memory. Five LSBs need to be appended to convert to a system address. This field is sized to address a 16 MiB Radar Memory. Accessing locations outside the range of the memory on the device will result in an error being generated by the Radar Memory module.
<b>RES</b>	27	r	<b>Reserved</b>
<b>TRRM</b>	28	rw	<b>Transient Removal Mode</b> Selects the replacement value for transient removal. $0_B$ <b>ZERO</b> , Set sample to zero $1_B$ <b>THRESHOLD</b> , Set to threshold Set sample value to TRTHRESH.THRESH*sign_of(value).
<b>TREN</b>	29	rw	<b>Transient Removal Enable</b> If set, samples with absolute values greater than the threshold will be replaced.
<b>EN</b>	30	rw	<b>MATH0 Enable</b> If set, the MATH0 Unit shall be enabled and DC offset will be removed. Other MATH0 functions also have separate enable bits in their control registers.
<b>BYPASS</b>	31	rw	<b>MATH0 Bypass</b> If set, bypass the MATH0 unit: input data will go directly to the MATH1 unit after DC and Transient Removal. The MATH0 unit will still receive input data and its functions will operate if enabled.
<b>0</b>	26:19	r	<b>Reserved</b> Reads as 0, shall be written with 0.

## Signal Processing Unit 2 (SPU2)

**Table 869 Access Mode Restrictions of M0CTRL sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BASE, BYPASS, EN, TREN, TRRM	
Otherwise (default)	r	BASE, BYPASS, EN, RES, TREN, TRRM	Default Access Mode (Bus Error on Write)

**Table 870 Reset Values of M0CTRL**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Transient Removal Threshold

This is an unsigned value that the absolute value of samples is compared with. If the absolute value is greater than the threshold it shall be replaced. If the sample value is 8000<sub>H</sub> it shall also be replaced. The absolute value of the real and imaginary components of complex data are taken and compared separately with the threshold. Replacement of a component is done based on the absolute value of that component only.

#### TRTHRESH

**Transient Removal Threshold** Reset Value: [Table 872](#)



Field	Bits	Type	Description
<b>THRESH</b>	14:0	rw	<b>Threshold</b>
<b>0</b>	31:15	r	<b>Reserved</b> Reads as 0, shall be written with 0.

**Table 871 Access Mode Restrictions of TRTHRESH sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	THRESH	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	THRESH	Default Access Mode (Bus Error on Write)

## Signal Processing Unit 2 (SPU2)

Table 872 Reset Values of **TRTHRESH**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Filter control**

Parameters controlling the behavior of the filters. There are two filters, one operating on the data stream in the forwards direction and the other operating in the backwards direction. All of the parameters apply to both filters except the CONJ field which only affects the backwards filter.

**FILTCTRL****Filter control**(00060<sub>H</sub>)Reset Value: [Table 874](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN</b>	<b>0</b>	<b>CONJ</b>			<b>0</b>						<b>NCOEFR</b>		<b>0</b>		
rw	r	rw			r						rw		r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					<b>0</b>						<b>NCOEFNR</b>				

Field	Bits	Type	Description
<b>NCOEFNR</b>	4:0	rw	<b>Number of non-recursive filter taps</b> The number of coefficients allocated to the non-recursive section of the filter. The rest of the coefficients are available to the recursive section. The valid range of values is 0 to 16 and values greater than 16 shall be treated as 16. If the sum of this field and NCOEFR is greater than 16 then NCOEFR takes priority and the number of non-recursive coefficients shall be 16-NCOEFR. If this field is set to zero then the unit is configured as a resonator and the input has no effect on the output. When ID_CONF2.FORMAT is set to COMPLEX the LSB shall be ignored and the number of complex coefficients has a valid range from 0 to 8 with greater values being treated as 8.
<b>NCOEFR</b>	22:18	rw	<b>Number of recursive filter taps</b> The number of coefficients allocated to the recursive section of the filter. The rest of the coefficients are available to the non-recursive section. The valid range of values is 0 to 16 and values greater than 16 shall be treated as 16. If the sum of this field and NCOEFNR is greater than 16 then this field takes priority and the number of non-recursive coefficients shall be 16-NCOEFNR. When ID_CONF2.FORMAT is set to COMPLEX the LSB shall be ignored and the number of complex coefficients has a valid range from 0 to 8 with greater values being treated as 8.

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
CONJ	29	rw	<b>Coefficient Conjugate Control</b> 0 <sub>B</sub> <b>OFF</b> , No conjugation 1 <sub>B</sub> <b>ON</b> , Conjugate coefficients The complex conjugate of the filter coefficients will be used in the backward filter.
EN	31	rw	<b>Filter Enable</b> If set, the filters are enabled otherwise the filters are bypassed and data passes directly to the next processing stage.
0	17:5, 28:23, 30	r	<b>Reserved</b> Reads as 0, shall be written with 0.

**Table 873 Access Mode Restrictions of FILTCTRL sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	CONJ, EN, NCOEFNR, NCOEFR	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	CONJ, EN, NCOEFNR, NCOEFR	Default Access Mode (Bus Error on Write)

**Table 874 Reset Values of FILTCTRL**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Filter coefficient

Each filter coefficient register has space for two 16-bit values called C0 and C1. In complex mode these are used as the real and imaginary parts of one coefficient. In real mode each field is a coefficient.

#### FILTCOEFc (c=0-7)

(00064 <sub>H</sub> +c*4)																Reset Value: <a href="#">Table 876</a>			
C1																			
C0																			

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
C0	15:0	rw	Real coefficient number $2*c$ or real part of the complex coefficient number c.
C1	31:16	rw	Real coefficient number $2*c+1$ or imaginary part of the complex coefficient number c.

**Table 875 Access Mode Restrictions of FILTCOEFc (c=0-7) sorted by descending priority**

Mode Name	Access Mode	Description
Master enabled in ACCEN	rw	C0, C1 Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	C0, C1 Default Access Mode (Bus Error on Write)

**Table 876 Reset Values of FILTCOEFc (c=0-7)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Detector control

Interference detector control.

#### DETCTRL

DETCTRL															
Detector control (00084 <sub>H</sub> )															
Reset Value: Table 878															
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16															
<b>HPFE N</b>															
rw R24 ALPHA															
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0															
R7 PCTHRESH WINSIZE OE EN															
r rw rw rw rw															

Field	Bits	Type	Description
EN	0	rw	<b>Enable</b> 0 <sub>B</sub> OFF, Detection function is disabled 1 <sub>B</sub> ON, Enable detection
OE	1	rw	<b>Output enable</b> If set, the output bit stream of the detector shall be written to Radar Memory. Has no effect if the EN==OFF

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>WINSIZE</b>	4:2	rw	<b>Size of detection window</b> The detection window size given as the number of samples either side of the central sample. Setting this field to zero means only the central cell is included in the detection decision.
<b>PCTHRESH</b>	6:5	rw	<b>Population count threshold</b> The detection window must have at least PCTHRESH+1 samples that exceed the threshold defined in DETCFG.THRESH for the detector output to be asserted. If PCTHRESH is set to zero just one sample needs to exceed the threshold for a detection to be made.
<b>R7</b>	15:7	r	<b>Reserved</b> Reads as 0, shall be written with 0.
<b>ALPHA</b>	23:16	rw	<b>High-pass filter coefficient</b> An unsigned fractional value. Setting this bit-field to FFFF <sub>H</sub> will configure the filter with the lowest possible cut-off frequency. The cut-off frequency increases as the value decreases.
<b>R24</b>	30:24	r	<b>Reserved</b> Reads as 0, shall be written with 0.
<b>HPFEN</b>	31	rw	<b>High-pass filter enable</b> 0 <sub>B</sub> OFF, No filtering 1 <sub>B</sub> ON, Filtering active

**Table 877 Access Mode Restrictions of DETCTRL sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	R24, R7	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	ALPHA, EN, HPFEN, OE, PCTHRESH, WINSIZE	
Otherwise (default)	r	ALPHA, EN, HPFEN, OE, PCTHRESH, R24, R7, WINSIZE	Default Access Mode (Bus Error on Write)

**Table 878 Reset Values of DETCTRL**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit 2 (SPU2)

### Detector Base Address Configuration

#### DETBASE

**Detector Base Address Configuration** **(00088<sub>H</sub>)** **Reset Value: Table 880**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														BASE	
r															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE														rw	

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> Base Address of where data shall be written to Radar Memory. This is a 32 byte (256 bit) aligned address relative to the start address of the Radar Memory. Five LSBs need to be appended to convert to a system address. This field is sized to address a 16 MiB Radar Memory. Accessing locations outside the range of the memory on the device will result in an error being generated by the Radar Memory module.
<b>0</b>	31:19	r	<b>Reserved</b> Reads as 0, shall be written with 0.

**Table 879 Access Mode Restrictions of DETBASE sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BASE	Default Access Mode (Bus Error on Write)

**Table 880 Reset Values of DETBASE**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit 2 (SPU2)

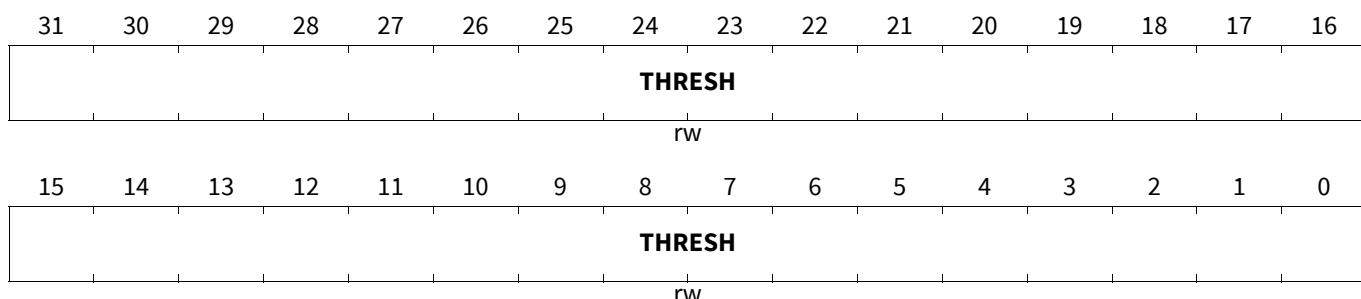
### Interference detector configuration

#### DETCFG

##### Interference detector configuration

(0008C<sub>H</sub>)

Reset Value: [Table 882](#)



Field	Bits	Type	Description
THRESH	31:0	rw	Threshold used for interference detection

**Table 881 Access Mode Restrictions of DETCFG sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	THRESH	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	THRESH	Default Access Mode (Bus Error on Write)

**Table 882 Reset Values of DETCFG**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Control of the replacement function

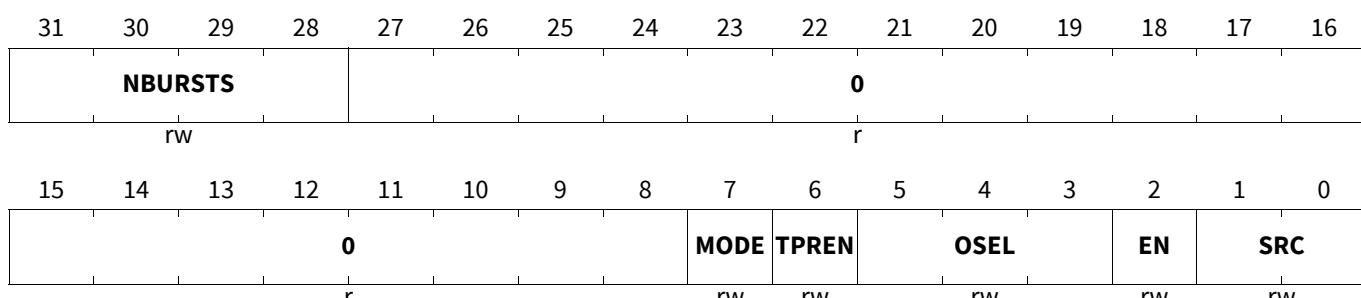
Options used to configure the replacement function.

#### RPLCTRL

##### Control of the replacement function

(00090<sub>H</sub>)

Reset Value: [Table 884](#)



## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>SRC</b>	1:0	rw	<p><b>Source of replacement values</b></p> <p><math>00_B</math> <b>ZERO</b>, Replace with zero</p> <p><math>01_B</math> <b>FILTER</b>, Use optionally tapered filter outputs</p> <p><math>10_B</math> <b>FORWARD</b>, Use output of the Forward filter</p> <p><math>11_B</math> <b>RESERVED</b>, Reserved for future replacement source</p>
<b>EN</b>	2	rw	<p><b>Enable</b></p> <p>If set, data samples shall be replaced in the locations defined by the BURSTb registers in SOFT mode or the input from the Detector in HARD mode with values according to the SRC field.</p>
<b>OSEL</b>	5:3	rw	<p><b>Output select</b></p> <p>Select what, if any, data is output from MATH0 to Radar Memory. Output to MATH1 is not affected. Output from the detector is controlled separately.</p> <p>When M0CTRL.MODE==COMPLEX 16 bit precision complex data shall be written otherwise 16 bit precision real data.</p> <p><math>000_B</math> <b>OFF</b>, No data output to memory</p> <p><math>001_B</math> <b>COMBINED</b>, Output combined data Data from replacement function is written to memory.</p> <p><math>010_B</math> <b>FILTER</b>, Output filtered data Data from the outputs of the filters are written to memory. For each sample, the value from the forward filter is written first followed by the value from the backward filter.</p> <p><math>011_B</math> <b>TRANS</b>, Output unfiltered data Data from after transient removal and before filtering are written to memory. Data will still be written even if both DC offset and transient removal are bypassed.</p> <p><math>100_B</math> <b>RES4</b>, Reserved</p> <p>...</p> <p><math>111_B</math> <b>RES7</b>, Reserved</p>
<b>TPREN</b>	6	rw	<p><b>Tapering Enable</b></p> <p>If set, the tapering function is enabled and replacement values are modified by the tapering value before being inserted into the output. Otherwise the values are inserted unmodified.</p>
<b>MODE</b>	7	rw	<p><b>Enable</b></p> <p><math>0_B</math> <b>SOFT</b>, Software controlled The NBURSTS bitfield and BURSTb registers control which samples the replacement function affects.</p> <p><math>1_B</math> <b>HARD</b>, Hardware controlled The detector output controls which samples the replacement function effects.</p>
<b>NBURSTS</b>	31:28	rw	<p><b>Number of Bursts</b></p> <p>Used when MODE==SOFT and defines the number of bursts that shall be replaced and the number of BURSTb registers that shall be read by the replacement function.</p> <p>Value must be in the range <math>0_D</math> to <math>8_D</math>. When NBURSTS==<math>0_D</math> no samples shall be replaced. The behavior is undefined for values outside this range.</p>

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
0	27:8	r	<b>Reserved</b> Reads as 0, shall be written with 0.

**Table 883 Access Mode Restrictions of RPLCTRL sorted by descending priority**

Mode Name	Access Mode	Description
Master enabled in ACCEN	rw	EN, MODE, NBURSTS, OSEL, SRC, TPREN Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	EN, MODE, NBURSTS, OSEL, SRC, TPREN Default Access Mode (Bus Error on Write)

**Table 884 Reset Values of RPLCTRL**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Burst descriptor

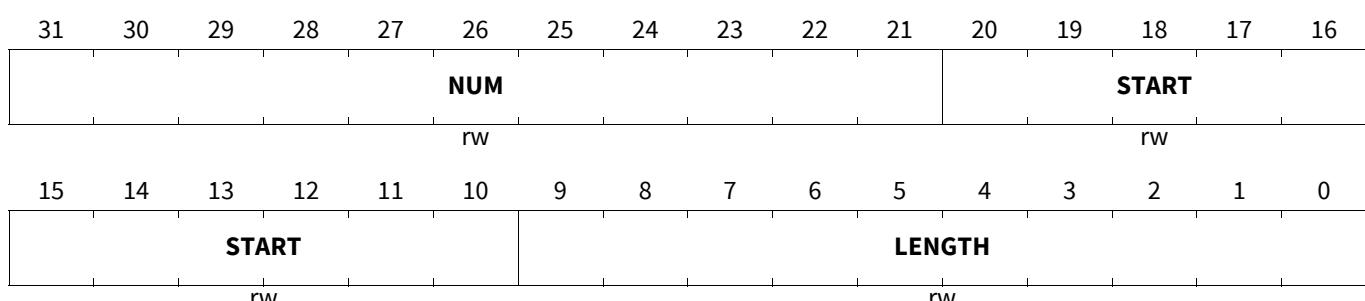
Interference burst ramp number, start bin and length. Used by the Combine Select and Taper Unit when the replacement function RPLCTRL.MODE==SOFT.

A burst is defined by the location and length of a burst. The location is given by the ramp number and sample number. The first ramp and first sample are both numbered zero.

Only the first RPLCTRL.NBURSTS registers shall have any effect.

#### BURST<sub>b</sub> (b=0-7)

**Burst descriptor** (00094<sub>H</sub>+b\*4) Reset Value: [Table 886](#)



Field	Bits	Type	Description
LENGTH	9:0	rw	<b>Length of Burst</b> The length of a burst is the value of this field plus one. The minimum length of a burst is one.
START	20:10	rw	<b>Start sample of the Burst</b>
NUM	31:21	rw	<b>Ramp Number of the Burst</b>

## Signal Processing Unit 2 (SPU2)

**Table 885 Access Mode Restrictions of BURSTb (b=0-7) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	LENGTH, NUM, START	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	LENGTH, NUM, START	Default Access Mode (Bus Error on Write)

**Table 886 Reset Values of BURSTb (b=0-7)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### DC offset sample count reciprocal

This register shall be programmed with the reciprocal of the number of samples as defined by the value of ID\_CONF.SMPLCNT encoded by calculating  $1/(ID\_CONF.SMPLCNT + 1)$  to 27 binary places. The number of leading zeros in the binary fractional value is the value of EXP and next 16 bits are the MANT value.

Otherwise setting the MANT bit-field to zero will disable DC offset removal by forcing the mean value to zero.

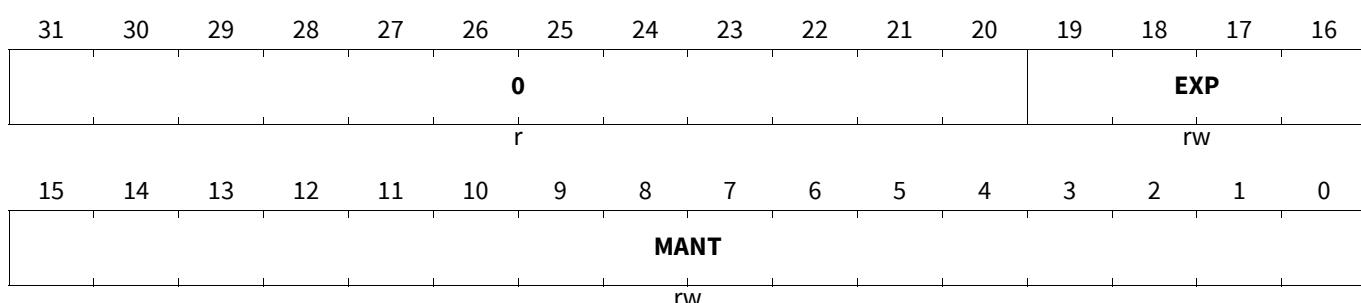
If MATH0 is disabled this register does not need to be configured and may be set to zero.

### SCRECIP

#### DC offset sample count reciprocal

(000B4<sub>H</sub>)

Reset Value: Table 888



Field	Bits	Type	Description
MANT	15:0	rw	Mantissa
EXP	19:16	rw	Exponent
0	31:20	r	Reserved

**Table 887 Access Mode Restrictions of SCRECIP sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	EXP, MANT	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	EXP, MANT	Default Access Mode (Bus Error on Write)

## Signal Processing Unit 2 (SPU2)

**Table 888 Reset Values of SCRECP**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Zero Insertion Mask

An array of registers that control the Zero Insertion function in the MATH1 unit.

Each bit in the mask corresponds to an FFT input sample. If a bit is set to DATA the data from the next input sample is inserted otherwise a zero value is inserted.

**ZI\_MASKn (n=0-7)**

**Zero Insertion Mask**

(000B8<sub>H</sub>+n\*4)

Reset Value: [Table 890](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SIOZ3 1	SIOZ3 0	SIOZ2 9	SIOZ2 8	SIOZ2 7	SIOZ2 6	SIOZ2 5	SIOZ2 4	SIOZ2 3	SIOZ2 2	SIOZ2 1	SIOZ2 0	SIOZ1 9	SIOZ1 8	SIOZ1 7	SIOZ1 6
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIOZ1 5	SIOZ1 4	SIOZ1 3	SIOZ1 2	SIOZ1 1	SIOZ1 0	SIOZ9	SIOZ8	SIOZ7	SIOZ6	SIOZ5	SIOZ4	SIOZ3	SIOZ2	SIOZ1	SIOZ0
rw															

Field	Bits	Type	Description
SIOZ <sub>i</sub> (i=0-31)	i	rw	<b>Select Input or Zero</b> <sub>0<sub>B</sub></sub> <b>ZERO</b> , Insert zero <sub>1<sub>B</sub></sub> <b>DATA</b> , Insert next input sample

**Table 889 Access Mode Restrictions of ZI\_MASKn (n=0-7) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw		Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r		Default Access Mode (Bus Error on Write)

**Table 890 Reset Values of ZI\_MASKn (n=0-7)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit 2 (SPU2)

### Loader Configuration

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Configuration Parameters for the Loader and MATH1 components of Streaming Processor 1

#### BEx\_LDR\_CONF (x=0-1)

##### Loader Configuration

(000D8<sub>H</sub>+x\*84<sub>H</sub>)

Reset Value: [Table 892](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>IFFT</b>	<b>FFTBY PS</b>	<b>FORMAT</b>			<b>SIZE</b>		<b>PHSHFT</b>					<b>EXPNT</b>			
rw	rw	rw			rw		rw					rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>DRPF</b>								<b>DRPL</b>							
rw								rw							

Field	Bits	Type	Description
<b>DRPL</b>	7:0	rw	<b>Drop Last</b> Stores a number "m". The last m samples of the input data will be ignored
<b>DRPF</b>	15:8	rw	<b>Drop First</b> Stores a number "n". The first n samples of the input data will be ignored
<b>EXPNT</b>	21:16	rw	<b>Common Exponent</b> For 16 <sub>D</sub> bit data, this indicates the alignment correction to be applied when reformatting to 32 bit. It should be set to 0 <sub>D</sub> if no adjustment of magnitude is needed. Otherwise the 16 <sub>D</sub> bit data will be shifted left by the number of bits set in this field and sign extended to 32 <sub>D</sub> bits. Empty bits at the right end of the 32 <sub>D</sub> bit value will be set to 0 <sub>B</sub> . The maximum expected value for this field is 16 <sub>D</sub> when using data read from Radar Memory as this is the largest value that will be generated using the compression function in the output data manager. Larger values can be used to create some gain in the input signal. Overflows in the shift operation will cause saturation of the output.
<b>PHSHFT</b>	23:22	rw	<b>Fast Phase Shift</b> This bitfield can set a fast phase shift of 0, 90, 180 or 270 degrees 00 <sub>B</sub> <b>ZERO</b> , No Phase Shift 01 <sub>B</sub> <b>NINETY</b> , 90 degree phase shift 10 <sub>B</sub> <b>ONEEIGHTY</b> , 180 degree phase shift 11 <sub>B</sub> <b>TWOSEVENTY</b> , 270 degree phase shift

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>SIZE</b>	27:24	rw	<b>FFT Size</b> Number of samples for the FFT. Must be a power of two between 8 and 2048. Unspecified values are reserved and must not be written. 0 <sub>H</sub> <b>8</b> , FFT is eight data samples 1 <sub>H</sub> <b>16</b> , FFT is sixteen data samples 2 <sub>H</sub> <b>32</b> , FFT is thirty-two data samples 3 <sub>H</sub> <b>64</b> , FFT is sixty-four data samples 4 <sub>H</sub> <b>128</b> , FFT is one hundred and twenty-eight data samples 5 <sub>H</sub> <b>256</b> , FFT is two hundred and fifty-six data samples 6 <sub>H</sub> <b>512</b> , FFT is five hundred and twelve data samples 7 <sub>H</sub> <b>1024</b> , FFT is one thousand and twenty-four data samples 8 <sub>H</sub> <b>2048</b> , FFT is two thousand and forty-eight data samples
<b>FORMAT</b>	29:28	rw	<b>Window Data Format</b> Format of the Window Function Data stored in the configuration memory. If a complex format is selected, the real component of each value is always stored at the lower address in memory. 00 <sub>B</sub> <b>REAL16</b> , 16 Bit Real Window Data 01 <sub>B</sub> <b>REAL32</b> , 32 Bit Real Window Data 10 <sub>B</sub> <b>CMPLX16</b> , 16 Bit Complex Window Data 11 <sub>B</sub> <b>CMPLX32</b> , 32 Bit Complex Window Data
<b>FTTBYP</b>	30	rw	<b>FFT Bypass</b> Bypass the FFT Module
<b>IFFT</b>	31	rw	<b>Inverse FFT Enable</b> If set, the accelerator will perform an IFFT operation instead of an FFT. In IFFT mode, the automatic descaling of the FFT output is disabled

**Table 891 Access Mode Restrictions of BEx\_LDR\_CONF (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	DRPF, DRPL, EXPNT, FTTBYP, FORMAT, IFFT, PHSHT, SIZE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	DRPF, DRPL, EXPNT, FTTBYP, FORMAT, IFFT, PHSHT, SIZE	Default Access Mode (Bus Error on Write)

**Table 892 Reset Values of BEx\_LDR\_CONF (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Loader Configuration Extended

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

## Signal Processing Unit 2 (SPU2)

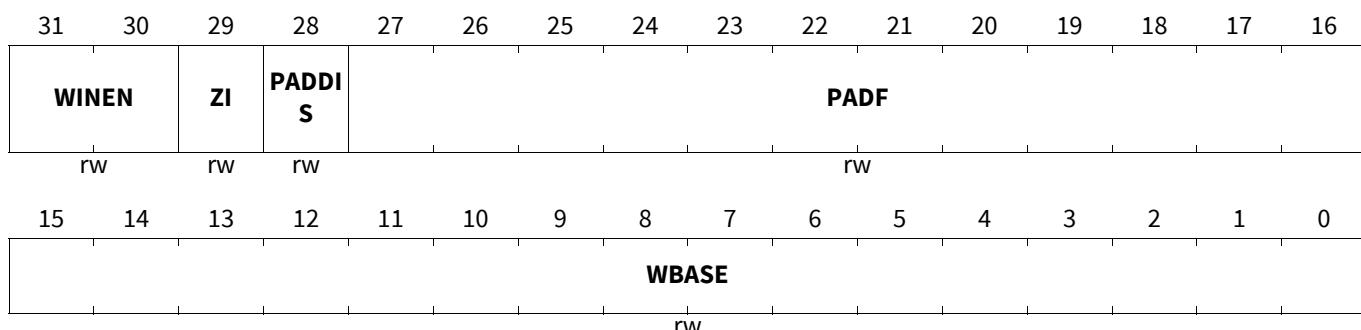
Configuration Parameters for the Loader and MATH1 components of Streaming Processor 1

### BEx\_LDR\_CONF2 (x=0-1)

Loader Configuration Extended

(000DC<sub>H</sub>+x\*84<sub>H</sub>)

Reset Value: [Table 894](#)



Field	Bits	Type	Description
WBASE	15:0	rw	<b>Window Coefficient Base Address</b> This is the base address of the Window Coefficient Array in the Configuration Memory. This address is relative to the Configuration Memory Base Address and must be 8 byte aligned. Any non-zero value written to bits[2:0] will be ignored.
PADF	27:16	rw	<b>Pad at Front</b>
PADDIS	28	rw	<b>Padding disable</b> When set, padding shall be disabled when LDR_CONF.FFTBYP is set. In this case PADF is ignored.
ZI	29	rw	<b>Zero Insertion Enable</b> When set, the zero insertion function shall be enabled.
WINEN	31:30	rw	<b>Window Function Enable</b> 00 <sub>B</sub> <b>OFF</b> , Disable Window Function 01 <sub>B</sub> <b>ACORREL</b> , Enable Autocorrelation Function Each input sample shall be multiplied by its complex conjugate. 10 <sub>B</sub> <b>WINDOW</b> , Enable Window Function 11 <sub>B</sub> <b>RESERVED</b> , Reserved

**Table 893 Access Mode Restrictions of BEx\_LDR\_CONF2 (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	PADDIS, PADF, WBASE, WINEN, ZI	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	PADDIS, PADF, WBASE, WINEN, ZI	Default Access Mode (Bus Error on Write)

## Signal Processing Unit 2 (SPU2)

Table 894 Reset Values of **BEx\_LDR\_CONF2 (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Antenna Offset

Offset Addresses for each antenna relative to the base address of the windowing coefficients. Each antenna has a unique offset stored as a 16 bit address. This address must be 8 byte aligned. Any non-zero value written to bits[2:0] will be ignored.

**BEx\_Aj\_ANTOFST (j=0-3;x=0-1)**

Antenna Offset <span style="float: right;">(000E0<sub>H</sub>+x*84<sub>H</sub>+j*4)</span>																Reset Value: <a href="#">Table 896</a>			
<b>ADROFST1_ANT</b>																			
rw																			
<b>ADROFST0_ANT</b>																			
rw																			

Field	Bits	Type	Description
<b>ADROFST<sub>i</sub>_AN</b>	16*i+15:16*	rw	<b>Antenna Offset Address</b>
T (i=0-1)	i		

Table 895 Access Mode Restrictions of **BEx\_Aj\_ANTOFST (j=0-3;x=0-1)** sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ADROFST <sub>i</sub> _ANT (i=0-1)	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ADROFST <sub>i</sub> _ANT (i=0-1)	Default Access Mode (Bus Error on Write)

Table 896 Reset Values of **BEx\_Aj\_ANTOFST (j=0-3;x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit 2 (SPU2)

### Unloader Configuration

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

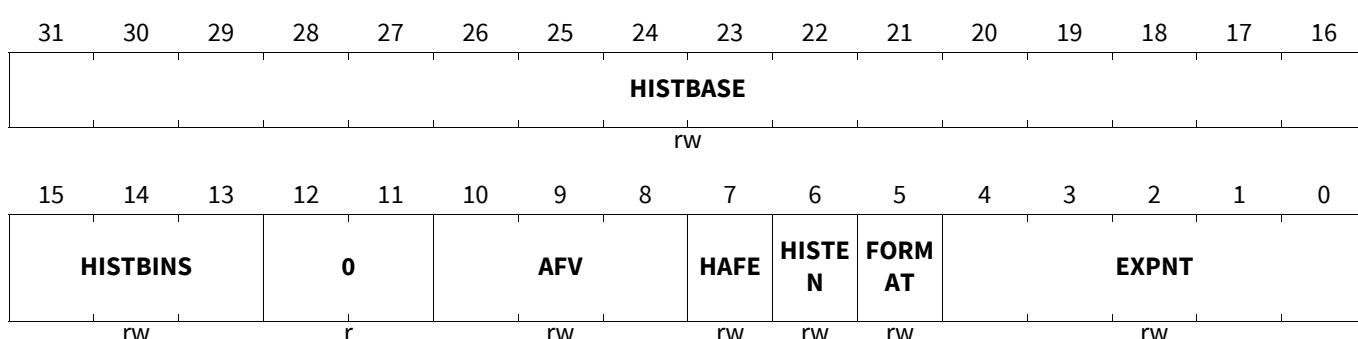
Configuration Parameters for the Power Histogram Unit contained in the Unloader. This register also defines the format of the data to be written to the buffer memory.

#### BEx\_UNLDR\_CONF (x=0-1)

##### Unloader Configuration

(000F0<sub>H</sub>+x\*84<sub>H</sub>)

Reset Value: [Table 898](#)



Field	Bits	Type	Description
<b>EXPNT</b>	4:0	rw	<p><b>Common Exponent</b></p> <p>When writing 16 bit precision data to the buffer memory, this field indicates the number of LSBs to be removed. i.e. the 32 bit data will be shifted right by this number of bit positions before truncating to the 16 least significant bits. In this mode, the value of the field should be restricted to values between 0<sub>D</sub> and 16<sub>D</sub>. The results of setting a value greater than 16<sub>D</sub> are undefined.</p> <p>When writing 32 bit precision data to the buffer memory, this field indicates the number of LSBs to be added. i.e. the 32 bit data will be shifted left by this number of bit positions.</p> <p>The remaining bits will be set to maximum or minimum value if overflow or underflow has occurred. The precision of the output data will be determined by the FORMAT bitfield.</p>
<b>FORMAT</b>	5	rw	<p><b>Input Data Format</b></p> <p>Required format of the data to be written to the buffer memory. Set to 1, 32 bit precision, after reset.</p> <p>0<sub>B</sub> <b>16BIT</b>, 16 Bit Complex Data            1<sub>B</sub> <b>32BIT</b>, 32 Bit Complex Data</p>
<b>HISTEN</b>	6	rw	<p><b>Histogram Enable</b></p> <p>If zero, no histogram is generated. If 1, then a power histogram is cumulatively assembled until the mode is disabled again</p>

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
HAFE	7	rw	<b>Histogram Antenna Filter Enable</b> Use this control bit to enable filtering of the data used to generate the power histogram based on antenna ID. If filtering is enabled only the selected antenna will contribute to the histogram data and the counter used for STRT and END check will only be incremented when an FFT associated that antenna ID is processed. 0 <sub>B</sub> <b>OFF</b> , Antenna filtering is disabled 1 <sub>B</sub> <b>ON</b> , Antenna filtering is enabled
AFV	10:8	rw	<b>Antenna Filter Value</b> If HAFE is set, this will be the antenna index to use to accumulate the power histogram data. Only data from this antenna index will be used to update the power histogram
HISTBINS	15:13	rw	<b>Number of power values per histogram bin.</b> Number of LSBs of power value to ignore when calculating histogram bin to increment. The calculated power value will be shifted right by the value of this bitfield and the resultant number used as the relative address of the histogram bin to increment.
HISTBASE	31:16	rw	<b>Histogram Base Address</b> Base address in the configuration RAM of the power histogram. This address is relative to the configuration RAM base address and must be 8 byte aligned. Any non-zero value written to bits[2:0] will be ignored.
0	12:11	r	<b>Reserved</b>

**Table 897 Access Mode Restrictions of BEx\_UNLDR\_CONF (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	AFV, EXPNT, FORMAT, HAFE, HISTBASE, HISTBINS, HISTEN	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	AFV, EXPNT, FORMAT, HAFE, HISTBASE, HISTBINS, HISTEN	Default Access Mode (Bus Error on Write)

**Table 898 Reset Values of BEx\_UNLDR\_CONF (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0020 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0020 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Unloader Configuration 2

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

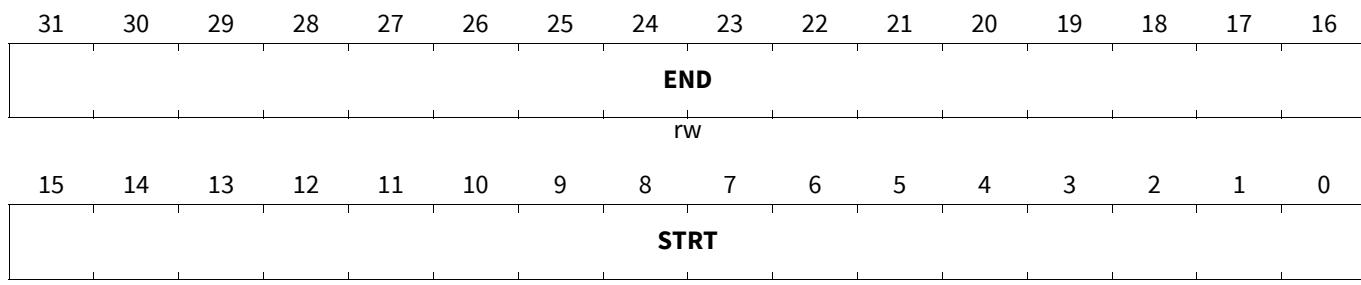
## Signal Processing Unit 2 (SPU2)

### BEx\_UNLDR\_CONF2 (x=0-1)

#### Unloader Configuration 2

( $000F4_H + x * 84_H$ )

Reset Value: [Table 900](#)



Field	Bits	Type	Description
STRT	15:0	rw	<b>Start Count</b> Delay from Start of Measurement Cycle before the accumulation of histogram is enabled. If set to 0 accumulation will start with the first FFT of the measurement cycle
END	31:16	rw	<b>End Count</b> Delay from Start of Measurement Cycle before the accumulation of histogram is ended. If set to all 1s accumulation will end with the final FFT of the measurement cycle

**Table 899 Access Mode Restrictions of BEx\_UNLDR\_CONF2 (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	END, STRT	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	END, STRT	Default Access Mode (Bus Error on Write)

**Table 900 Reset Values of BEx\_UNLDR\_CONF2 (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Unloader Configuration 3

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

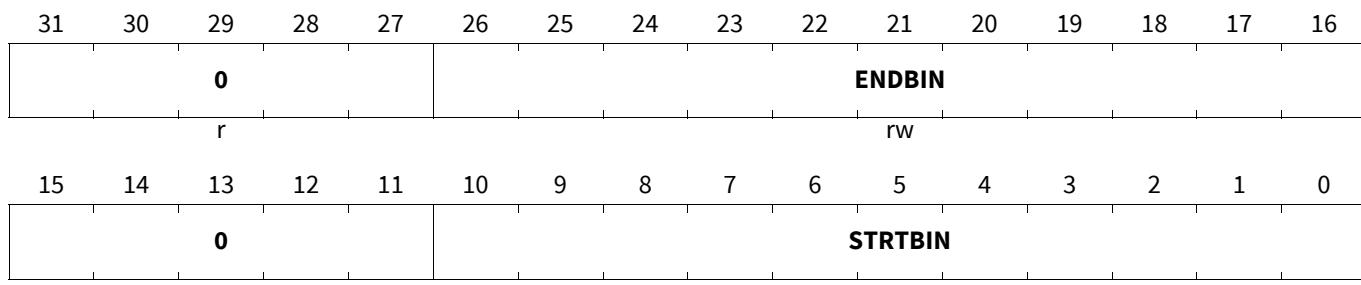
## Signal Processing Unit 2 (SPU2)

### BEx\_UNLDR\_CONF3 (x=0-1)

#### Unloader Configuration 3

( $000F8_H + x^*84_H$ )

Reset Value: [Table 902](#)



Field	Bits	Type	Description
STRTBIN	10:0	rw	<b>Start FFT Bin</b> Bin in each FFT result generated where histogram calculation starts. For calculating the histogram only bins between STRTBIN and ENDBIN (inclusive) will be included in the histogram. FFT bin numbering starts with 0.
ENDBIN	26:16	rw	<b>End FFT Bin</b> Bin in each FFT result generated where histogram calculation starts. For calculating the histogram only bins between STRTBIN and ENDBIN (inclusive) will be included in the histogram. FFT bin numbering starts with 0. Setting this field to a number greater than the last bin in each FFT will be the same as setting it to the last bin of the FFT.
0	15:11, 31:27	r	<b>RESERVED</b>

**Table 901 Access Mode Restrictions of BEx\_UNLDR\_CONF3 (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ENDBIN, STRTBIN	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ENDBIN, STRTBIN	Default Access Mode (Bus Error on Write)

**Table 902 Reset Values of BEx\_UNLDR\_CONF3 (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Spare Configuration Register

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

## Signal Processing Unit 2 (SPU2)

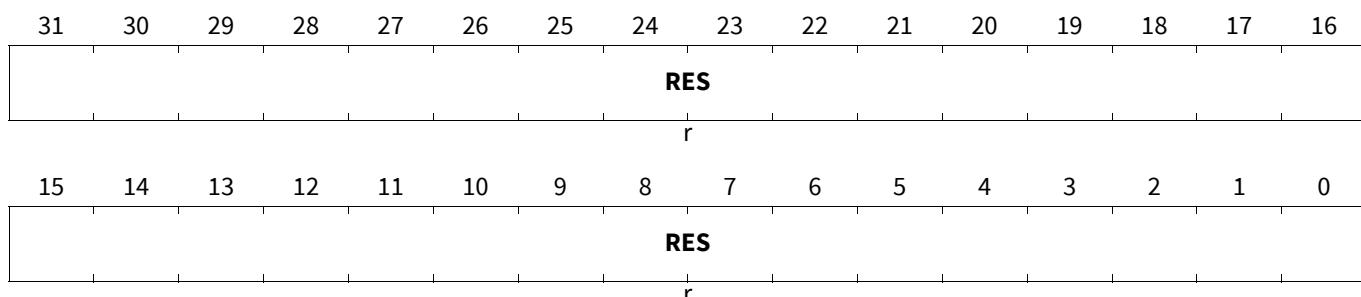
Reserved for future expansion of the Unloader functionality

### BEx\_UNLDR\_ACFG (x=0-1)

Spare Configuration Register

(000FC<sub>H</sub>+x\*84<sub>H</sub>)

Reset Value: [Table 904](#)



Field	Bits	Type	Description
RES	31:0	r	Reserved

**Table 903 Access Mode Restrictions of BEx\_UNLDR\_ACFG (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)

**Table 904 Reset Values of BEx\_UNLDR\_ACFG (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Output Data Processor Base Write Address

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

This register configures the base address that the Output Data Processor uses for writing FFT results to the Radar Memory.

## Signal Processing Unit 2 (SPU2)

### BEx\_ODP\_BASE (x=0-1)

Output Data Processor Base Write Address ( $00100_H + x * 84_H$ )

Reset Value: [Table 906](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														BASE	
r														rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE														rw	

Field	Bits	Type	Description
BASE	18:0	rw	<b>Radar Memory Base Address</b> The base address to be used when writing data to Radar Memory. This is a word (256 bit) address relative to the Radar Memory base address. All writes will be 32 byte words. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing an FFT, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Accessing locations outside the range of the memory on the device will result in an error being generated by the Radar Memory module.
0	31:19	r	<b>RESERVED</b>

**Table 905 Access Mode Restrictions of BEx\_ODP\_BASE (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BASE	Default Access Mode (Bus Error on Write)

**Table 906 Reset Values of BEx\_ODP\_BASE (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Output Data Processor Configuration

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

This register configures the Output Data Processor for writing FFT results to the Radar Memory.

## Signal Processing Unit 2 (SPU2)

## BEx\_ODP\_CONF (x=0-1)

Output Data Processor Configuration

(00104<sub>H</sub>+x\*84<sub>H</sub>)

Reset Value: Table 908

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				COMP RESS	HPFP	ROF	IPF	EXPNT				SCALE	FTR	MODE	
r				rw	rw	rw	rw	rw				rw	rw	rw	

Field	Bits	Type	Description
<b>MODE</b>	0	rw	<b>ODP Mode</b> Major Operating Mode for the ODP 0 <sub>B</sub> <b>OFF</b> , FFT Output Disable The FFT output path is disabled. 1 <sub>B</sub> <b>ON</b> , FFT On The FFT data output path is enabled
<b>FTR</b>	1	rw	<b>Force to Real</b> The complex component of the FFT output data will be set to zero when it is read from the buffer memory This will affect all procesing operations of the MATH2 unit.
<b>SCALE</b>	2	rw	<b>Scale Results to 16 bit</b> If set, the results will be scaled to 16 bit precision before writing to EMEM. The scaling factor used will be from the ODP_CONF.EXPNT bitfield 0 <sub>B</sub> <b>32BIT</b> , Output is writeen with 32 bit precision 1 <sub>B</sub> <b>16BIT</b> , Output is written with 16 bit precision
<b>EXPNT</b>	7:3	rw	<b>Common Exponent</b> When writing 16 bit precision data to the Radar Memory, this field indicates the number of LSBs to be removed. i.e. the 32 bit data will be shifted right by this number of bit positions before truncating to the 16 least significant bits. The remaining bits will be set to maximum or minimum value if overflow or underflow has occurred
<b>IPF</b>	8	rw	<b>In Place FFT</b> When set, the Output Data Processor will attempt to write FFT results to the memory locations freed up by the reading of the input data.
<b>ROF</b>	9	rw	<b>Real Only Format</b> When set, the Output Data Processor will write FFT output data in a real only format. The imaginary component of the results will not be written. This mode differs from the function of "Force to Real" (FTR) because any other calculations performed using the FFT data will use the imaginary component. If consistency is required between the real only data written and the results of the other channels, then FTR should also be set.

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>HPFP</b>	10	rw	<b>Half Precision Floating Point</b> When set, the Output Data Processor will write FFT output data in half precision floating point format. If this bit is set then SCALE is redundant and will be ignored. FTR and ROF can still be used.
<b>COMPRESS</b>	11	rw	<b>Compress FFT Data</b> If set and the HPFP bit is also set then data will be packed into 256 bit words in a compressed form.
<b>0</b>	31:12	r	<b>Reserved</b>

**Table 907 Access Mode Restrictions of BEx\_ODP\_CONF (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	COMPRESS, EXPNT, FTR, HPFP, IPF, MODE, ROF, SCALE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	COMPRESS, EXPNT, FTR, HPFP, IPF, MODE, ROF, SCALE	Default Access Mode (Bus Error on Write)

**Table 908 Reset Values of BEx\_ODP\_CONF (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### ODP Inner Loop Address Offset

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Options used to configure the Output DMA for writing data to memory when using "In Place FFT" mode. Register field descriptions should be read in conjunction with the In Place FFT section of the User Manual where usage and permissible combinations are more fully defined

#### BEx\_ODP\_ILO (x=0-1)

**ODP Inner Loop Address Offset** (00108<sub>H</sub>+x\*84<sub>H</sub>) Reset Value: [Table 910](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										ILO					
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ILO										0					
r															

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
ILO	23:5	rw	<b>Inner Loop Offset</b> The Inner Loop Offset is a word address. The smallest permissible increment is equivalent to the size of a single word in EMEM (i.e. 32 bytes). The field is aligned in the register so that a byte address (the offset address from the EMEM base address) is returned when the register is read.
0	4:0, 31:24	r	<b>Reserved</b>

**Table 909 Access Mode Restrictions of BEx\_ODP\_ILO (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ILO	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ILO	Default Access Mode (Bus Error on Write)

**Table 910 Reset Values of BEx\_ODP\_ILO (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### ODP Outer Loop Address Offset

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

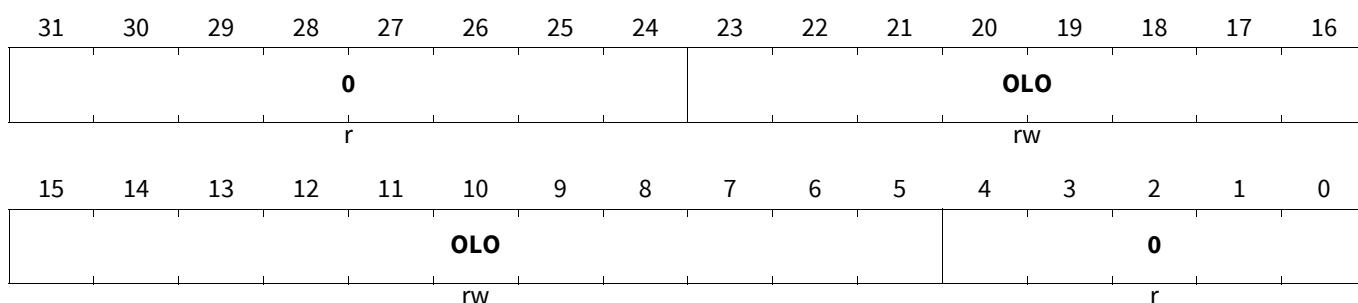
Options used to configure the Output DMA for writing data to memory when using "In Place FFT" mode. Register field descriptions should be read in conjunction with the In Place FFT section of the User Manual where usage and permissible combinations are more fully defined

#### BEx\_ODP\_OLO (x=0-1)

##### ODP Outer Loop Address Offset

(0010C<sub>H</sub>+x\*84<sub>H</sub>)

Reset Value: [Table 912](#)



## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
OLO	23:5	rw	<b>Outer Loop Offset</b> The Outer Loop Offset is a word address. The smallest permissible increment is equivalent to the size of a single word in EMEM (i.e. 32 bytes). The field is aligned in the register so that a byte address (the offset address from the EMEM base address) is returned when the register is read.
0	4:0, 31:24	r	<b>Reserved</b>

**Table 911 Access Mode Restrictions of BEx\_ODP\_OLO (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	OLO	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	OLO	Default Access Mode (Bus Error on Write)

**Table 912 Reset Values of BEx\_ODP\_OLO (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### ODP Bin Offset Address Configuration

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

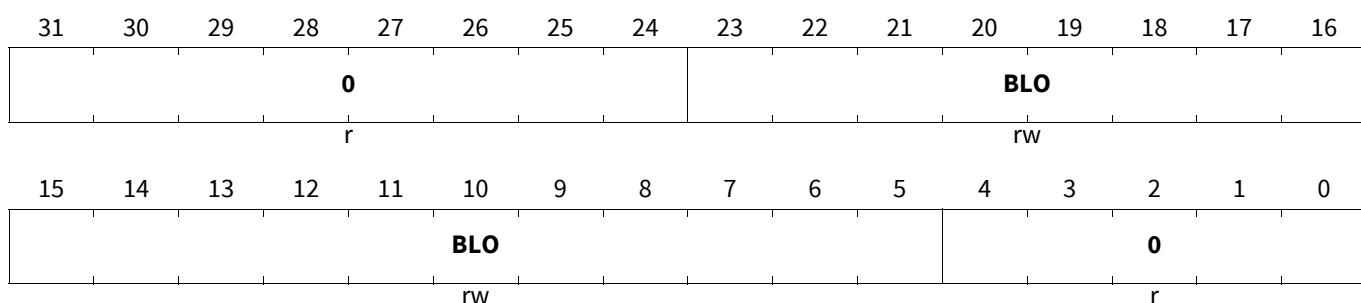
Options used to configure the Output DMA for writing data to memory when using "In Place FFT" mode. Register field descriptions should be read in conjunction with the In Place FFT section of the User Manual where usage and permissible combinations are more fully defined

#### BEx\_ODP\_BLO (x=0-1)

##### ODP Bin Offset Address Configuration

(00110<sub>H</sub>+x\*84<sub>H</sub>)

Reset Value: [Table 914](#)



## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>BLO</b>	23:5	rw	<b>Bin Loop Offset</b> The Bin Loop Offset is a word address. The smallest permissible increment is equivalent to the size of a single word in EMEM (i.e. 32 bytes). The field is aligned in the register so that a byte address (the offset address from the EMEM base address) is returned when the register is read.
<b>0</b>	4:0, 31:24	r	<b>Reserved</b>

**Table 913 Access Mode Restrictions of [BEx\\_ODP\\_BLO \(x=0-1\)](#) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BLO	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BLO	Default Access Mode (Bus Error on Write)

**Table 914 Reset Values of [BEx\\_ODP\\_BLO \(x=0-1\)](#)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### ODP Inner and Outer Loop Repeat

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Options used to configure the Output DMA for writing data to memory when using "In Place FFT" mode. Register field descriptions should be read in conjunction with the In Place FFT section of the User Manual where usage and permissible combinations are more fully defined

#### **BEx\_ODP\_IOLR (x=0-1)**

#### ODP Inner and Outer Loop Repeat

(00114<sub>H</sub>+x\*84<sub>H</sub>)

Reset Value: [Table 916](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								OLR							
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ILR							
r															

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
ILR	12:0	rw	<b>Inner Loop Repeat</b> Repetition Count for the Inner Address Loop. The loop will repeat at least once. The overall number of repetitions will be the value of this bitfield + 1.
OLR	28:16	rw	<b>Outer Loop Repeat</b> Repetition Count for the Outer Address Loop. The loop will repeat at least once. The overall number of repetitions will be the value of this bitfield + 1.
0	15:13, 31:29	r	<b>Reserved</b>

**Table 915 Access Mode Restrictions of BEx\_ODP\_IOLR (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ILR, OLR	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ILR, OLR	Default Access Mode (Bus Error on Write)

**Table 916 Reset Values of BEx\_ODP\_IOLR (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### ODP Bin Loop Repeat

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

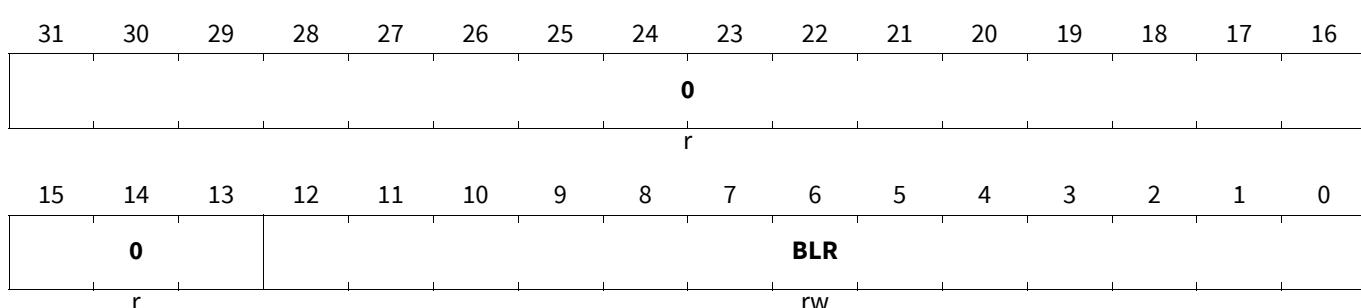
Options used to configure the Output DMA for writing data to memory when using "In Place FFT" mode. Register field descriptions should be read in conjunction with the In Place FFT section of the User Manual where usage and permissible combinations are more fully defined

#### BEx\_ODP\_BLR (x=0-1)

##### ODP Bin Loop Repeat

(00118<sub>H</sub>+x\*84<sub>H</sub>)

Reset Value: [Table 918](#)



## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
BLR	12:0	rw	<b>Bin Loop Repeat</b> Repetition Count for the Bin Address Loop. The loop will repeat at least once. The overall number of repetitions will be the value of this bitfield + 1.
0	31:13	r	<b>Reserved</b>

**Table 917 Access Mode Restrictions of BEx\_ODP\_BLR (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BLR	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BLR	Default Access Mode (Bus Error on Write)

**Table 918 Reset Values of BEx\_ODP\_BLR (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### NCI DBF Base Address Configuration

Base address for writing the integration results (NCI or DBF) data to Memory.

#### BEx\_ND\_BASEb (b=0-1;x=0-1)

**NCI DBF Base Address Configuration (0011C<sub>H</sub>+x\*84<sub>H</sub>+b\*4)**

**Reset Value: Table 920**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN</b>							<b>0</b>								<b>BASE</b>
rw															
15    14    13    12    11    10    9    8    7    6    5    4    3    2    1    0															
<b>BASE</b>															
rw															

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> The base address to be used when writing data to Radar Memory. It is a word (256 bit) address relative to the Radar Memory base address. All writes will be 32 byte words. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing an FFT, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Accessing locations outside the range of the memory on the device will result in an error being generated by the Radar Memory module.
<b>EN</b>	31	rw	<b>Enable output</b> If set, output data to Radar Memory.
<b>0</b>	30:19	r	<b>Reserved</b> Reads as 0, shall be written with 0.

Table 919 Access Mode Restrictions of **BEx\_ND\_BASEb (b=0-1;x=0-1)** sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE, EN	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BASE, EN	Default Access Mode (Bus Error on Write)

Table 920 Reset Values of **BEx\_ND\_BASEb (b=0-1;x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Dual Integration Unit Control

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Controls the operation of the Dual Integration Units providing the coherent (DBF) and non-coherent integration (NCI) functions.

## Signal Processing Unit 2 (SPU2)

## BEx\_NDCTRL (x=0-1)

Dual Integration Unit Control

(00124<sub>H</sub>+x\*84<sub>H</sub>)Reset Value: [Table 922](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DBFRE AL	SUMO NLY1														
rw	rw				rw						rw		rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Field	Bits	Type	Description
MODE	2:0	rw	<p><b>NCI Enable</b>            Operating mode for the Integration units            000<sub>B</sub> <b>OFF</b>, Both Units Disabled            001<sub>B</sub> <b>NCI_OFF</b>, Unit 0 Does NCI, Unit 1 Disabled            010<sub>B</sub> <b>NCI_NCI</b>, Both Units do NCI            011<sub>B</sub> <b>NCI_DBF</b>, Unit 0 does NCI, Unit 1 does DBF            100<sub>B</sub> <b>DBF_OFF</b>, Unit 0 does DBF, Unit 1 Disabled            101<sub>B</sub> <b>DBF_DBF</b>, Both Units do DBF            110<sub>B</sub> <b>RES6</b>, Reserved            111<sub>B</sub> <b>RES7</b>, Reserved         </p>

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>FORMAT<sub>i</sub> (i=0-1)</b>	14*i+5:14*i+3	rw	<p><b>Output Format</b>  Sets the precision of the output data written to the Radar Memory.</p> <p><b>000<sub>B</sub> REAL16BIT</b>, 16 Bit Data Output (real)  When the unit is in DBF mode and DBFREAL==COMPLEX the linear power value shall be written to memory otherwise the real result of the calculation. When the unit is in NCI mode linear power shall be written to memory.</p> <p><b>001<sub>B</sub> REAL32BIT</b>, 32 Bit Data Output (real)  When the unit is in DBF mode and DBFREAL==COMPLEX the linear power value shall be written to memory otherwise the real result of the calculation. When the unit is in NCI mode linear power shall be written to memory. This data may be read by the SPU using RM_CONF2.FORMAT==PWR32BIT</p> <p><b>010<sub>B</sub> CMPLX16BIT</b>, 16 Bit Precision Complex Data Output  When the unit is in DBF mode 16 bit precision, complex data will be written. When the unit is in NCI mode 16 bit precision, complex data will be written with the linear power value in the real component and a zero imaginary component.</p> <p><b>011<sub>B</sub> CMPLX32BIT</b>, 32 Bit Precision Complex Data Output  When the unit is in DBF mode, 32 bit precision, complex data will be written. When the unit is in NCI mode 32 bit precision, complex data will be written with the linear power value in the real component and a zero imaginary component.</p> <p><b>100<sub>B</sub> RHPFP</b>, Real Half Precision Floating Point  When the unit is in DBF mode and DBFREAL==COMPLEX the linear power value shall be written to memory otherwise the real result of the calculation. When the unit is in NCI mode linear power shall be written to memory. This data may be read by the SPU using RM_CONF2.FORMAT==PWR16FP</p> <p><b>101<sub>B</sub> CHPFP</b>, Complex Half Precision Floating Point  When the unit is in NCI mode, half precision floating point, complex data will be written with the linear power value in the real component and a zero imaginary component.</p> <p><b>110<sub>B</sub> RES6</b>, Reserved</p> <p><b>111<sub>B</sub> RES7</b>, Reserved</p>
<b>SCALE<sub>i</sub> (i=0-1)</b>	14*i+7:14*i+6	rw	<p><b>Result Scaling</b>  This bitfield controls the scaling factor applied to the Integration results. The scaling factor can be used to prevent an overflow of the results.</p> <p><b>00<sub>B</sub> OFF</b>, No Rescaling</p> <p><b>01<sub>B</sub> DIV2</b>, Divide Result by 2</p> <p><b>10<sub>B</sub> DIV4</b>, Divide Result by 4</p> <p><b>11<sub>B</sub> DIV8</b>, Divide Result by 8</p>
<b>USEANT<sub>i</sub> (i=0-1)</b>	14*i+15:14*i+8	rw	<p><b>Antennae to Use</b>  This field allows the application to select which antennae are to be used in Integration. Each bit in the field corresponds to one antenna with the LSB mapping to antenna 0 and the MSB to antenna 7. If a bit is set to 1, then the relevant antenna will be included in the weighted sum.</p>

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>SUMONLYi (i=0-1)</b>	14*i+16	rw	<p><b>Only do sum, no multiplication</b></p> <p>If set, no multiplication is done in the Integration, in effect, making all weights equal to unity. No weights are read from the Configuration Memory.</p>
<b>DBFREAL</b>	31	rw	<p><b>Real Value</b></p> <p>Digital Beam Forming calculations using Complex Data will use the real component only. The imaginary component will be set to zero before the calculation. No imaginary component will be written to memory.</p> <p>Has no effect when unit is in NCI mode.</p> <p>0<sub>B</sub> <b>COMPLEX</b>, Complex Arithmetic Calculations will use real and imaginary components</p> <p>1<sub>B</sub> <b>REAL</b>, Real Arithmetic Only Calculations will use the real component of the input data only. The imaginary component will be discarded. Data written to memory will be a real number</p>

**Table 921 Access Mode Restrictions of BEx\_NDCTRL (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	DBFREAL, FORMATi (i=0-1), MODE, SCALEi (i=0-1), SUMONLYi (i=0-1), USEANTI (i=0-1)	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	DBFREAL, FORMATi (i=0-1), MODE, SCALEi (i=0-1), SUMONLYi (i=0-1), USEANTI (i=0-1)	Default Access Mode (Bus Error on Write)

**Table 922 Reset Values of BEx\_NDCTRL (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### NCI and DBF Coefficient Base address

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Base addresses in Configuration Memory of the Non-Coherent Integration and Digital Beam Forming Coefficients.

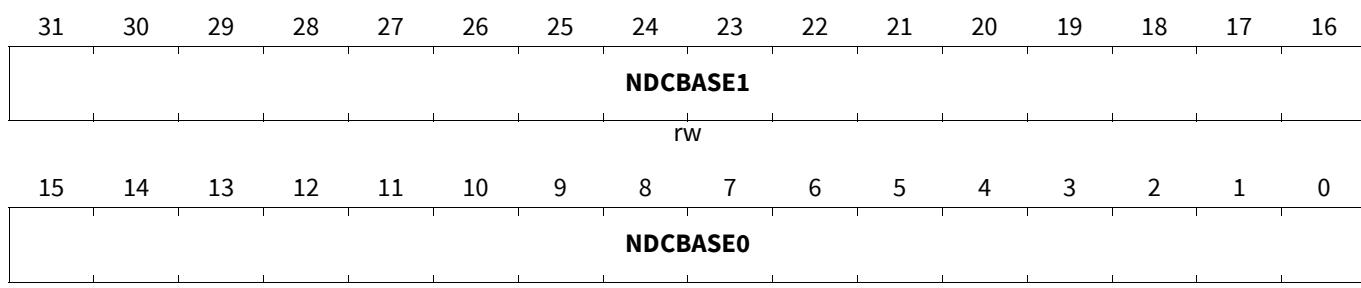
## Signal Processing Unit 2 (SPU2)

### BEx\_NDCBASE (x=0-1)

NCI and DBF Coefficient Base address

(00128<sub>H</sub>+x\*84<sub>H</sub>)

Reset Value: [Table 924](#)



Field	Bits	Type	Description
NDCBASEi (i=0-1)	16*i+15:16*	rw	<b>NCI DBF Coefficient Base Address</b> This is the base address of the Intergration Coefficient Array in the Configuration Memory. This address is relative to the Configuration Memory Base Address and must be 8 byte aligned. Any non-zero value written to bits[2:0] will be ignored.

**Table 923 Access Mode Restrictions of BEx\_NDCBASE (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw		Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r		Default Access Mode (Bus Error on Write)

**Table 924 Reset Values of BEx\_NDCBASE (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Power Information Channel Control

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Enables the Output DMA channel writing log<sub>2</sub> power information to the Radar Memory and sets the base address for writes.

## Signal Processing Unit 2 (SPU2)

## BEx\_PWRCTRL (x=0-1)

Power Information Channel Control

(0012C<sub>H</sub>+x\*84<sub>H</sub>)Reset Value: [Table 926](#)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN							0									BASE
rw							r									rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	BASE
																rw

Field	Bits	Type	Description
BASE	18:0	rw	<b>Radar Memory Base Address</b> The base address to be used when writing data to Radar Memory. This is a word (256 bit) address relative to the Radar Memory base address. All writes will be 32 byte words. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing an FFT, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Accessing locations outside the range of the memory on the device will result in an error being generated by the Radar Memory module.
EN	31	rw	<b>Enable</b> Enables the writing of Signal Power to Radar Memory on a per FFT bin basis. The data is a 16 bit precision representation of log2 power.
0	30:19	r	<b>Reserved</b> Reads as 0, shall be written with 0.

**Table 925 Access Mode Restrictions of BEx\_PWRCTRL (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE, EN	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BASE, EN	Default Access Mode (Bus Error on Write)

**Table 926 Reset Values of BEx\_PWRCTRL (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit 2 (SPU2)

## CFAR Module Control

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

This register controls the operating mode of the CFAR module and the Local Maximum Unit.

## BEx\_CFARCTRL (x=0-1)

## CFAR Module Control

(00130<sub>H</sub>+x\*84<sub>H</sub>)Reset Value: [Table 928](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		0	CFAR_GOSE	CFAR_CAE	EXTNSN			SEWIN					LFUNC		
r		rw	rw	rw	rw			rw					rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		LFUNC		SRCSEL	LBLMODE	LBLCON	LCLMAX	LFOSEL					0		
		rw		rw	rw	rw	rw	rw	rw	rw	rw	rw		r	

Field	Bits	Type	Description
<b>LFOSEL</b>	3	rw	<b>Local Maximum or Logical Function Output Select</b> Select whether the LCLMAX or Logical Function is written to Radar Memory. A single Output DMA port is shared between Local Maximum and Logical Function output.  0 <sub>B</sub> <b>LCLMAX</b> , Local Maximum Output 1 <sub>B</sub> <b>LFUNC</b> , Logical Function Output
<b>LCLMAX</b>	4	rw	<b>Local Maximum Engine Enable</b> Select whether the LCLMAX Engine is enabled. 0 <sub>B</sub> <b>OFF</b> , Local Maximum Unit Off 1 <sub>B</sub> <b>ON</b> , Local Maximum Unit On
<b>LBLCON</b>	6:5	rw	<b>Label Content Select</b> Select the content of the Label List output. If this field is set to OFF and LBLMODE==INT or LBLMODE==NOINT then no data shall be output. 00 <sub>B</sub> <b>OFF</b> , No output 01 <sub>B</sub> <b>ADDR</b> , Output Address Only Output the 32 bit system address corresponding to the location of the detected peak in Radar Memory. 10 <sub>B</sub> <b>INDEX</b> , Output 3D Index Only Output a 32 bit word composed of the bin number (bits 12:0), FFT number (bits 25:13) and channel number (bits 28:26) corresponding to the bin location of the detected peak. When detection is performed on integrated data (and the DMA is in Integration Mode) the channel number is fixed at zero. The 2 MSBs are always zero. 11 <sub>B</sub> <b>BOTH</b> , Output Address and Index Output the system address followed by the 3D index.

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>LBLMODE</b>	8:7	rw	<p><b>Label Output Mode</b></p> <p>This bitfield selects the operating mode for target Label List output. The Label List output function, when configured to do so, writes a 32 bit word to Memory</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> <b>OFF</b>, No Output or Counting</li> <li>01<sub>B</sub> <b>COUNT</b>, Count Only The target counter shall be incremented each time there is a Label Event.</li> <li>10<sub>B</sub> <b>NOINT</b>, Output and Count, No Interrupt The target counter shall be incremented and the label written to memory. No interrupt shall be generated.</li> <li>11<sub>B</sub> <b>INT</b>, Output, Count and Interrupt The target counter LBLCNT.CNT shall be incremented, the label written to memory and an interrupt generated according to the counter value and the limit value CFARCFG3.LBLOLM.</li> </ul>
<b>SRCSEL</b>	10:9	rw	<p><b>CFAR Source Selection</b></p> <p>Input source selection for CFAR and Local Maximum units.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> <b>FFT</b>, Use FFT Data</li> <li>01<sub>B</sub> <b>INT0</b>, Use Data from Integration Unit 0 The relevant Integration unit must be configured to be active.</li> <li>10<sub>B</sub> <b>INT1</b>, Use Data from Integration Unit 1 The relevant Integration unit must be configured to be active.</li> <li>11<sub>B</sub> <b>RES</b>, Reserved</li> </ul>
<b>LFUNC</b>	18:11	rw	<p><b>Logical Function</b></p> <p>This bit field defines the logical function used to combine the three outputs from GOS-CFAR, CA-CFAR and LCLMAX units. The 8 bits form a one bit wide look-up table with 8 locations. The locations are addressed using an address constructed as follows: address(0) = LCLMAX, address(1) = GOS-CFAR and address(2) = CA-CFAR. The output of the function will be the bit value found at the addressed location in the register. For example, if bit 6 of this field is set then the output will be logical one only when the outputs of GOS-CFAR and CA-CFAR are one and LCLMAX is zero, otherwise the output will be zero.</p>
<b>SEWIN</b>	24:19	rw	<p><b>Spectrum Extension Window</b></p> <p>This bitfield defines the window size to be used for spectrum extension used for all detectors, CFAR and Local Maximum. The window size should be set to the maximum required by the active detectors and CFAR algorithm or algorithms. This field must not be set to a non-zero value unless at least one of the CFAR or Local Maximum Units is enabled.</p>

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>EXTNSN</b>	26:25	rw	<b>Spectrum Extension</b> Enable Spectrum Extension in either range or velocity modes for the CFAR or Local Maximum Units. This field must be set to "OFF" unless either at least one of the CFAR units or the Local Maximum Unit is enabled. 00 <sub>B</sub> <b>OFF</b> , No Spectrum Extension 01 <sub>B</sub> <b>RANGE</b> , Range Spectrum Extension 10 <sub>B</sub> <b>VELOCITY</b> , Velocity Spectrum Extension 11 <sub>B</sub> <b>Reserved</b> , Reserved
<b>CFAR_CAE</b>	27	rw	<b>CFAR CA Engine Enable</b> Select whether the CFAR CA Engine is enabled. 0 <sub>B</sub> <b>OFF</b> , CA Engine Off 1 <sub>B</sub> <b>ON</b> , CA Engine On
<b>CFAR_GOSE</b>	28	rw	<b>CFAR GOS Engine Enable</b> Select whether the CFAR GOS Engine is enabled. 0 <sub>B</sub> <b>OFF</b> , GOS Engine Off 1 <sub>B</sub> <b>ON</b> , GOS Engine On
<b>0</b>	2:0, 31:29	r	<b>Reserved</b> Reads as 0, shall be written with 0.

**Table 927 Access Mode Restrictions of BEx\_CFARCTRL (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	CFAR_CAE, CFAR_GOSE, EXTNSN, LBLCON, LBLMODE, LCLMAX, LFOSEL, LFUNC, SEWIN, SRCSEL	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	CFAR_CAE, CFAR_GOSE, EXTNSN, LBLCON, LBLMODE, LCLMAX, LFOSEL, LFUNC, SEWIN, SRCSEL	Default Access Mode (Bus Error on Write)

**Table 928 Reset Values of BEx\_CFARCTRL (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### CFAR CA Engine Base Address

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Base address for writing the CA-CFAR results data to Memory

## Signal Processing Unit 2 (SPU2)

**BEx\_CABASE (x=0-1)**

CFAR CA Engine Base Address

(00134<sub>H</sub>+x\*84<sub>H</sub>)Reset Value: [Table 930](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN</b>						<b>0</b>									<b>BASE</b>
rw						r									rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>BASE</b>															
rw															

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> Base Address in the Radar Memory to be used for writing the CA-CFAR information. All writes will be 32 byte words. This is a word (256 bit) address and 5 LSBs need to be added to translate this to an address in the system memory map. All writes will be 32 byte words. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing a measurement cycle, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Accessing locations outside the range of the memory on the device will result in an error being generated by the Radar Memory module.
<b>EN</b>	31	rw	<b>Enable output</b> If set, output data to Radar Memory.
<b>0</b>	30:19	r	<b>Reserved</b> Reads as 0, shall be written with 0.

**Table 929 Access Mode Restrictions of BEx\_CABASE (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE, EN	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BASE, EN	Default Access Mode (Bus Error on Write)

**Table 930 Reset Values of BEx\_CABASE (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit 2 (SPU2)

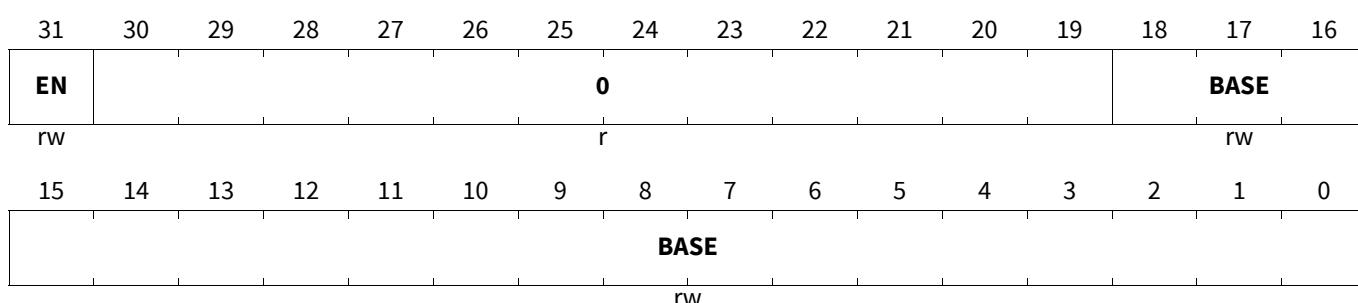
## CFAR GOS Engine Base Address

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Base address for writing the GOS-CFAR results data to Memory

## BEx\_GOSBASE (x=0-1)

## CFAR GOS Engine Base Address

(00138<sub>H</sub>+x\*84<sub>H</sub>)Reset Value: [Table 932](#)

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> Base Address in the Radar Memory to be used for writing the GOS-CFAR information. All writes will be 32 byte words. This is a word (256 bit) address and 5 LSBs need to be added to translate this to an address in the system memory map. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing a measurement cycle, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Accessing locations outside the range of the memory on the device will result in an error being generated by the Radar Memory module.
<b>EN</b>	31	rw	<b>Enable output</b> If set, output data to Radar Memory.
<b>0</b>	30:19	r	<b>Reserved</b> Reads as 0, shall be written with 0.

Table 931 Access Mode Restrictions of BEx\_GOSBASE (x=0-1) sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE, EN	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BASE, EN	Default Access Mode (Bus Error on Write)

## Signal Processing Unit 2 (SPU2)

Table 932 Reset Values of BEx\_GOSBASE (x=0-1)

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Local Maximum Detection Base Address

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Base address for writing the Local Maximum or Logical Function results data to Memory

## BEx\_LCLMAXBASE (x=0-1)

Local Maximum Detection Base Address (0013C<sub>H</sub>+x\*84<sub>H</sub>)

Reset Value: Table 934

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN</b>						<b>0</b>									<b>BASE</b>
rw						r									rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>BASE</b>															
rw															

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> Base Address in the EMEM to be used for writing the Local Maximum information. All writes will be 32 byte words. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing a measurement cycle, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Accessing locations outside the range of the memory on the device will result in an error being generated by the Radar Memory module.
<b>EN</b>	31	rw	<b>Enable output</b> If set, output data to Radar Memory.
<b>0</b>	30:19	r	<b>Reserved</b> Reads as 0, shall be written with 0.

## Signal Processing Unit 2 (SPU2)

**Table 933 Access Mode Restrictions of BEx\_LCLMAXBASE (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE, EN	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BASE, EN	Default Access Mode (Bus Error on Write)

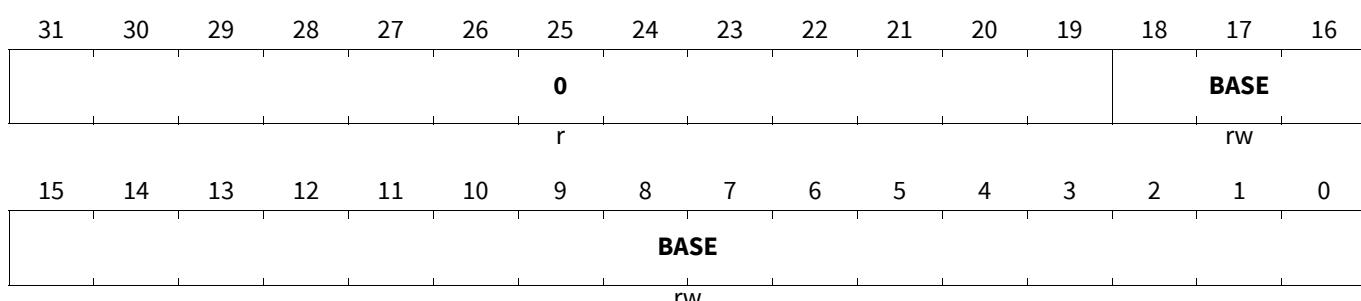
**Table 934 Reset Values of BEx\_LCLMAXBASE (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Label List Base Address**

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Base address for writing target index label data to Memory

**BEx\_LBLBASE (x=0-1)****Label List Base Address**(00140<sub>H</sub>+x\*84<sub>H</sub>)Reset Value: [Table 936](#)

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> Base Address in the EMEM to be used for writing the Label information. All writes will be 32 byte words. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing a measurement cycle, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Accessing locations outside the range of the memory on the device will result in an error being generated by the Radar Memory module.
<b>0</b>	31:19	r	<b>Reserved</b> Reads as 0, shall be written with 0.

## Signal Processing Unit 2 (SPU2)

**Table 935 Access Mode Restrictions of BEx\_LBLBASE (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BASE	Default Access Mode (Bus Error on Write)

**Table 936 Reset Values of BEx\_LBLBASE (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

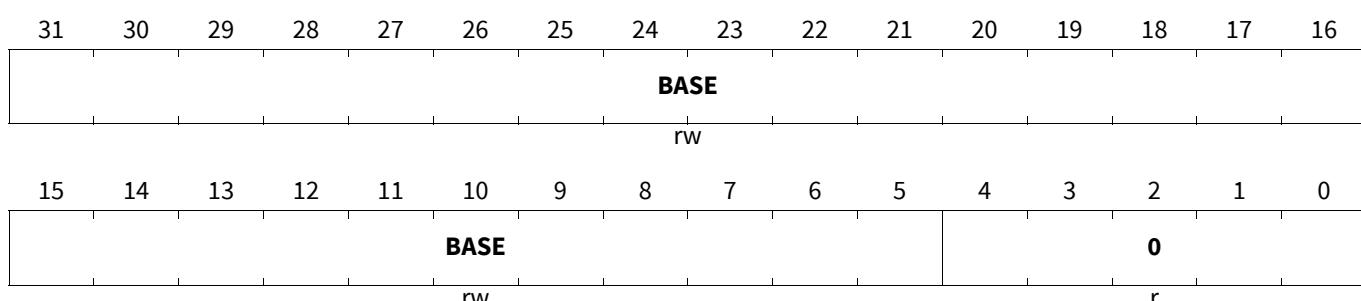
### Label address calculation base address

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

The base address used in the calculation of label addresses.

#### BEx\_LBLCALC (x=0-1)

**Label address calculation base address (00144<sub>H</sub>+x\*84<sub>H</sub>)**      **Reset Value: Table 938**



Field	Bits	Type	Description
<b>BASE</b>	31:5	rw	<b>Label address caculation base address</b> The base address to be used when calculating label address data. This forms a 256-bit word system address when the 5 zero-value LSBs of the register are considered.
<b>0</b>	4:0	r	<b>Reserved</b>

**Table 937 Access Mode Restrictions of BEx\_LBLCALC (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BASE	Default Access Mode (Bus Error on Write)

## Signal Processing Unit 2 (SPU2)

**Table 938 Reset Values of BEx\_LBLCALC (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Label X multiplier

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Parameter used to calculate address part in the labelling function.

**BEx\_LBLX (x=0-1)**

**Label X multiplier**

(00148<sub>H</sub>+x\*84<sub>H</sub>)

Reset Value: [Table 940](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							0							MUL	
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUL															
rw															

Field	Bits	Type	Description
<b>MUL</b>	18:0	rw	<b>X multiplier</b> Parameter used to calculate address part in the labelling function.
<b>0</b>	31:19	r	<b>Reserved</b>

**Table 939 Access Mode Restrictions of BEx\_LBLX (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	MUL	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	MUL	Default Access Mode (Bus Error on Write)

**Table 940 Reset Values of BEx\_LBLX (x=0-1)**

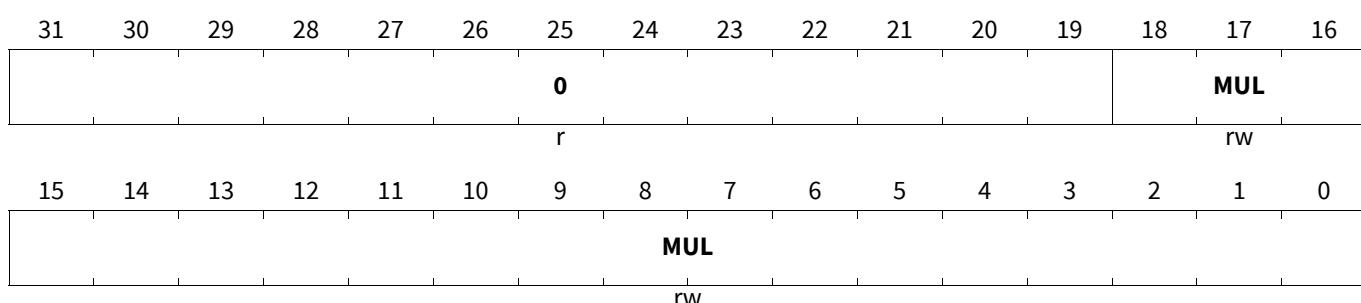
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit 2 (SPU2)

**Label Y multiplier**

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Parameter used to calculate address part in the labelling function.

**BEx\_LBLY (x=0-1)****Label Y multiplier**(0014C<sub>H</sub>+x\*84<sub>H</sub>)Reset Value: [Table 942](#)

Field	Bits	Type	Description
<b>MUL</b>	18:0	rw	<b>Y Multiplier</b> Parameter used to calculate address part in the labelling function.
<b>0</b>	31:19	r	<b>Reserved</b>

**Table 941 Access Mode Restrictions of BEx\_LBLY (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw MUL		Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r MUL		Default Access Mode (Bus Error on Write)

**Table 942 Reset Values of BEx\_LBLY (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Label Z multiplier**

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Parameter used to calculate address part in the labelling function.

## Signal Processing Unit 2 (SPU2)

## BEx\_LBLZ (x=0-1)

Label Z multiplier

(00150<sub>H</sub>+x\*84<sub>H</sub>)Reset Value: [Table 944](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								MUL							
r								rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUL								rw							

Field	Bits	Type	Description
MUL	18:0	rw	Z multiplier Parameter used to calculate address part in the labelling function.
0	31:19	r	Reserved

**Table 943 Access Mode Restrictions of BEx\_LBLZ (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw MUL		Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r MUL		Default Access Mode (Bus Error on Write)

**Table 944 Reset Values of BEx\_LBLZ (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Label repeat values for X and Y directions**

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

Parameters used to calculate address part in the labelling function.

## Signal Processing Unit 2 (SPU2)

**BEx\_LBL (x=0-1)**Label repeat values for X and Y directions (00154<sub>H</sub>+x\*84<sub>H</sub>)Reset Value: [Table 946](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
															<b>YRPT</b>
r	0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
r	0														
															<b>XRPT</b>

Field	Bits	Type	Description
<b>XRPT</b>	12:0	rw	<b>X repeat</b> The number of elements in the X direction shall be the value of this bitfield + 1.
<b>YRPT</b>	28:16	rw	<b>Y repeat</b> The number of elements in the Y direction shall be the value of this bitfield + 1.
<b>0</b>	15:13, 31:29	r	<b>Reserved</b>

**Table 945 Access Mode Restrictions of BEx\_LBL (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	XRPT, YRPT	
Otherwise (default)	r	XRPT, YRPT	

**Table 946 Reset Values of BEx\_LBL (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Sideband Control**

This register is duplicated. The second register instance is used when a second processing pass is made on the same set of data or different options are needed on alternate sets of data. The second processing pass is controlled by writing to the DPASS\_CONF.EN and DPASS\_CONF.SWITCH bitfields

This register configures and enables the statistical information sideband channel.

## Signal Processing Unit 2 (SPU2)

**BEx\_SBCTRL (x=0-1)**

## Sideband Control

(00158<sub>H</sub>+x\*84<sub>H</sub>)Reset Value: [Table 948](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN</b>						<b>0</b>									<b>BASE</b>
rw						r									rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>BASE</b>															
rw															rw

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Radar Memory Base Address</b> The base address to be used when writing data to Radar Memory. This is a word (256 bit) address relative to the Radar Memory base address. All writes will be 32 byte words. Data will be buffered internally until 32 bytes are available. The generated address will automatically increment after each write. If a partial word remains at the end of processing a measurement cycle, the data will be padded to 32 bytes and flushed to Radar Memory. This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. Accessing locations outside the range of the memory on the device will result in an error being generated by the Radar Memory module.
<b>EN</b>	31	rw	<b>Enable</b> Enable to Power Analysis Sideband Channel.
<b>0</b>	30:19	r	<b>Reserved</b>

**Table 947 Access Mode Restrictions of BEx\_SBCTRL (x=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE, EN	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	BASE, EN	Default Access Mode (Bus Error on Write)

**Table 948 Reset Values of BEx\_SBCTRL (x=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Bin Rejection Mask**

Every bit in these registers corresponds to an FFT sample point. If the bit is cleared, then the corresponding sample point will either be removed from the data written to EMEM or set to a zero value

## Signal Processing Unit 2 (SPU2)

This register contains part of the bin rejection mask. Each bit corresponds to a bin of the FFT output. If the bit is cleared, then the corresponding bin will be filtered from the output.

### BINm\_REJ (m=0-63)

#### Bin Rejection Mask

(001E0<sub>H</sub>+m\*4)

Reset Value: [Table 950](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>B_R31</b>	<b>B_R30</b>	<b>B_R29</b>	<b>B_R28</b>	<b>B_R27</b>	<b>B_R26</b>	<b>B_R25</b>	<b>B_R24</b>	<b>B_R23</b>	<b>B_R22</b>	<b>B_R21</b>	<b>B_R20</b>	<b>B_R19</b>	<b>B_R18</b>	<b>B_R17</b>	<b>B_R16</b>
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>B_R15</b>	<b>B_R14</b>	<b>B_R13</b>	<b>B_R12</b>	<b>B_R11</b>	<b>B_R10</b>	<b>B_R9</b>	<b>B_R8</b>	<b>B_R7</b>	<b>B_R6</b>	<b>B_R5</b>	<b>B_R4</b>	<b>B_R3</b>	<b>B_R2</b>	<b>B_R1</b>	<b>B_R0</b>
rw															

Field	Bits	Type	Description
<b>B_Rn (n=0-31)</b>	n	rw	<b>BIN</b> Set to "1" to allow the corresponding bin through the unit

**Table 949 Access Mode Restrictions of BINm\_REJ (m=0-63) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	B_Rn (n=0-31)	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	B_Rn (n=0-31)	Default Access Mode (Bus Error on Write)

**Table 950 Reset Values of BINm\_REJ (m=0-63)**

Reset Type	Reset Value	Note
Application Reset	FFFF FFFF <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	FFFF FFFF <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

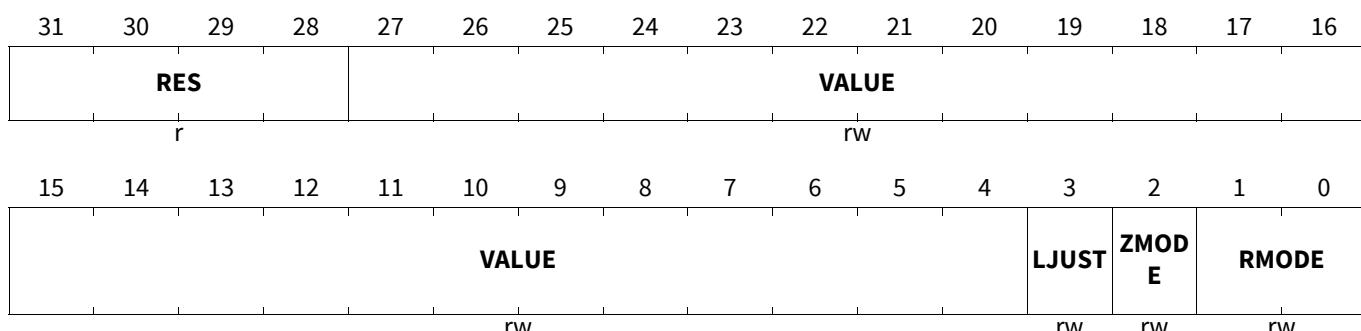
### Bin Rejection Unit Control

Control Parameters for the Bin Rejection Unit. The Bin Rejection Unit takes inputs from a 2048 set of registers (one bit per FFT data point) and data from the CFAR and Threshold Unit. These bits are used to zero bins or remove them from the outputs data stream. Additionally bins can be compared to a threshold value and set to zero if the threshold is exceeded. The threshold value is set as a magnitude. The Magnitude Approximation of the complex data bins is used for the comparison.

## Signal Processing Unit 2 (SPU2)

**BINREJCTRL**

## Bin Rejection Unit Control

(002E0<sub>H</sub>)Reset Value: [Table 952](#)

Field	Bits	Type	Description
<b>RMODE</b>	1:0	rw	<b>Bin Rejection Mode</b> 00 <sub>B</sub> <b>OFF</b> , Unit Disabled Pass all data 01 <sub>B</sub> <b>REJ</b> , Bin Rejection remove the selected bins from the output data 10 <sub>B</sub> <b>ZERO</b> , Zero set the selected bins to zero 11 <sub>B</sub> <b>RESERVED</b> , Reserved
<b>ZMODE</b>	2	rw	<b>Threshold Rejection Mode</b> 0 <sub>B</sub> <b>OFF</b> , Unit Disabled Pass all data 1 <sub>B</sub> <b>ZETH</b> , Zero with Threshold Set bins to zero if they exceed the programmed threshold (BINREJCTRL.VALUE)
<b>LJUST</b>	3	rw	<b>Left Justify</b> The 24 bit VALUE bitfield is compared against a 32 bit value. This field controls whether the comparison is made against the 24 MSBs or 24 LSBs 0 <sub>B</sub> <b>RIGHT</b> , Right Justify. Pad with 8 MSBs to create a 32 bit comparison value 1 <sub>B</sub> <b>LEFT</b> , Left Justify. Pad with 8 LSBs to create a 32 bit comparison value
<b>VALUE</b>	27:4	rw	<b>Threshold Value</b> This is the threshold value to be used for comparison. This is a 24 bit magnitude value.
<b>RES</b>	31:28	r	<b>Reserved</b>

**Table 951 Access Mode Restrictions of BINREJCTRL sorted by descending priority**

Mode Name	Access Mode		Description	
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error	
	rw	LJUST, RMODE, VALUE, ZMODE		
Otherwise (default)	r	LJUST, RES, RMODE, VALUE, ZMODE		
		Default Access Mode (Bus Error on Write)		

## Signal Processing Unit 2 (SPU2)

**Table 952 Reset Values of BINREJCTRL**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

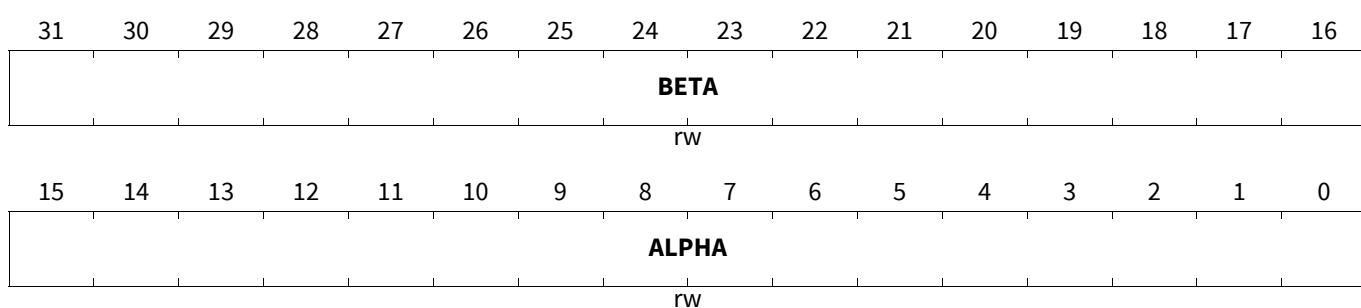
### Magnitude Approximation Constants

#### MAGAPPROX

##### Magnitude Approximation Constants

(002E4<sub>H</sub>)

Reset Value: [Table 954](#)



Field	Bits	Type	Description
<b>ALPHA</b>	15:0	rw	<b>Alpha Constant</b> Alpha value for Magnitude Approximation Algorithm
<b>BETA</b>	31:16	rw	<b>Beta Constant</b> Beta Value for Magnitude Approximation Algorithm

**Table 953 Access Mode Restrictions of MAGAPPROX sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	ALPHA, BETA	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	ALPHA, BETA	Default Access Mode (Bus Error on Write)

**Table 954 Reset Values of MAGAPPROX**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

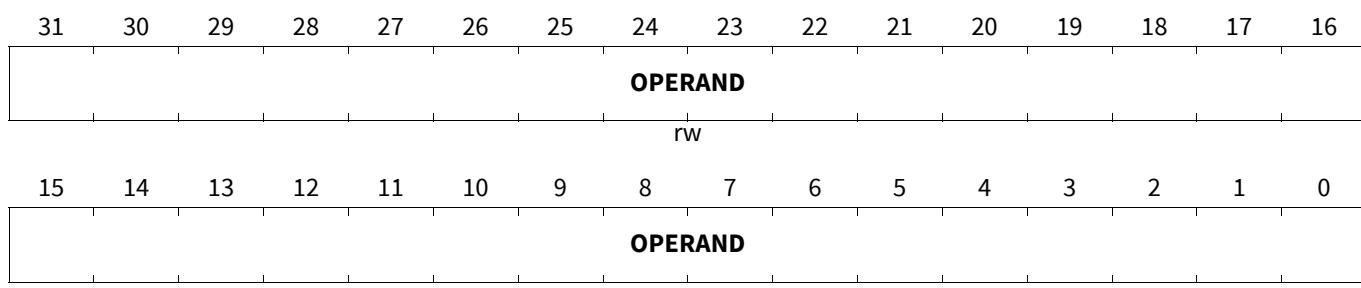
### Scalar Addition Operand

Parameter to be used when adjusting the FFT output data

## Signal Processing Unit 2 (SPU2)

### SCALARADD

#### Scalar Addition Operand

(002E8<sub>H</sub>)Reset Value: [Table 956](#)

Field	Bits	Type	Description
<b>OPERAND</b>	31:0	rw	<b>Operand for Scaling</b> Additive Scaling factor for FFT output data. This is 32bit signed integer value which will be added to both the real and imaginary components of the FFT output data.

**Table 955 Access Mode Restrictions of SCALARADD sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	OPERAND	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	OPERAND	Default Access Mode (Bus Error on Write)

**Table 956 Reset Values of SCALARADD**

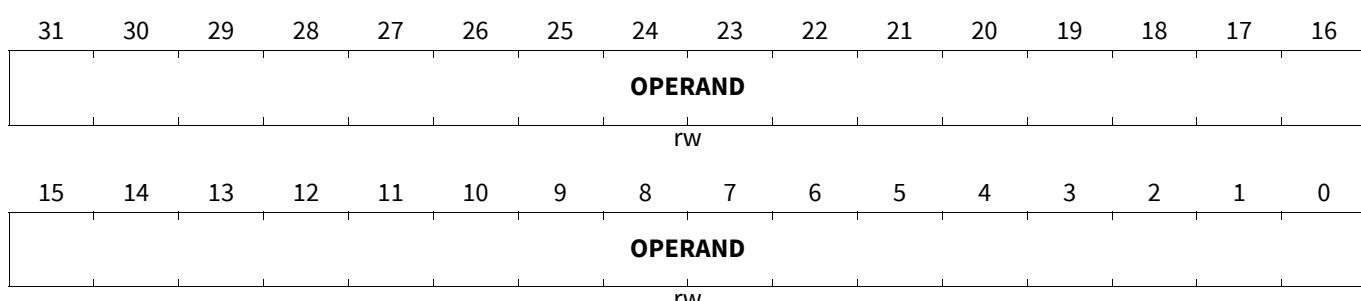
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Scalar Multiplication Operand

Parameter to be used when adjusting the FFT output data

### SCALARMULT

#### Scalar Multiplication Operand

(002EC<sub>H</sub>)Reset Value: [Table 958](#)

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>OPERAND</b>	31:0	rw	<b>Operand for Scaling</b> This is a multiplicative scaling factor to be applied to the FFT output data. It is a 32bit signed integer value with a nominal range of -1 to +1. Both the real and imaginary components are multiplied by the number.

**Table 957 Access Mode Restrictions of SCALARMULT sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	OPERAND	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	OPERAND	Default Access Mode (Bus Error on Write)

**Table 958 Reset Values of SCALARMULT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### CFAR Configuration

Configuration Parameters for the integrated CFAR module are stored in this register

#### CFARCFG1

##### CFAR Configuration

(002F0<sub>H</sub>)

Reset Value: [Table 960](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CABETA															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		CAWINCELL				CAGUARD						GOSALGO		CAALGO	
r			rw				rw					rw		rw	

Field	Bits	Type	Description
<b>CAALGO</b>	1:0	rw	<b>CA-CFAR Algorithm Select</b> Select the CA-CFAR algorithm 00 <sub>B</sub> <b>CASHCFAR</b> , CASH-CFAR CASH-CFAR algorithm 01 <sub>B</sub> <b>CACFAR</b> , CA-CFAR 10 <sub>B</sub> <b>CAGOCFAR</b> , CAGO-CFAR 11 <sub>B</sub> <b>CASOCFAR</b> , CASO-CFAR

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>GOSALGO</b>	3:2	rw	<b>GOS-CFAR Algorithm Select</b> Select the CA-CFAR algorithm 00 <sub>B</sub> <b>GOSCA</b> , GOSCA-CFAR GOSCA-CFAR Algorithm 01 <sub>B</sub> <b>GOSGOCFAR</b> , GOSGO-CFAR GOSGO-CFAR Algorithm 10 <sub>B</sub> <b>GOSSOCFAR</b> , GOSSO-CFAR GOSSO-CFAR Algorithm 11 <sub>B</sub> <b>RES</b> , Deprecated second GOSSO-CFAR setting Reserved (if set will use GOSSO-CFAR Algorithm)
<b>CAGUARD</b>	9:4	rw	<b>Guard Cells</b> Number of guard cells in CA-CFAR leading and lagging the cell under test
<b>CAWINCELL</b>	12:10	rw	<b>Window Cells Exponent</b> Exponent of 2 for defining the number of active cells in leading/lagging windows to be averaged in CA-CFAR. 2^(WINCELL) should be less than or equal to 32.
<b>CABETA</b>	31:16	rw	<b>CA-CFAR Beta</b> Additive constant scaling the CA-CFAR threshold
<b>0</b>	15:13	r	<b>Reserved</b>

**Table 959 Access Mode Restrictions of CFARCFG1 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	CAALGO, CABETA, CAGUARD, CAWINCELL, GOSALGO	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	CAALGO, CABETA, CAGUARD, CAWINCELL, GOSALGO	Default Access Mode (Bus Error on Write)

**Table 960 Reset Values of CFARCFG1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### CFAR Configuration 2

Configuration Parameters for the integrated CFAR module are stored in this register

## Signal Processing Unit 2 (SPU2)

## CFARCFG2

## CFAR Configuration 2

(002F4<sub>H</sub>)Reset Value: [Table 962](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES								CASHWIN				GOSWINCELL			
				r				rw				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDXLG								IDXLD				GOSGUARD			
				rw				rw				rw			

Field	Bits	Type	Description
<b>GOSGUARD</b>	5:0	rw	<b>Guard Cells</b> Number of guard cells in GOS-CFAR leading and lagging the cell under test
<b>IDXLD</b>	10:6	rw	<b>Index Lead</b> Index of sorted statistic in leading window in GOS-CFAR
<b>IDXLG</b>	15:11	rw	<b>Index Lag</b> Index of sorted statistic in lagging window in GOS-CFAR
<b>GOSWINCELL</b>	21:16	rw	<b>Window Cells Exponent</b> Value defining the number of active cells in leading/lagging windows to be averaged in GOS-CFAR. WINCELL should be less than or equal to 32. This sets a window size of between 1 and 32 cells. A value of 0 or a value greater than 32 should not be programmed. Behaviors in these cases is undefined
<b>CASHWIN</b>	24:22	rw	<b>CASH Subwindow</b> Exponent of 2 for defining the number of active cells in leading/lagging subwindow when using the CASH algorithm in the CA-CFAR engine. $2^{(CASHWIN)}$ should be less than or equal to 32.
<b>RES</b>	31:25	r	<b>Reserved</b>

**Table 961 Access Mode Restrictions of CFARCFG2 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	CASHWIN, GOSGUARD, GOSWINCELL, IDXLD, IDXLG	
Otherwise (default)	r	CASHWIN, GOSGUARD, GOSWINCELL, IDXLD, IDXLG, RES	Default Access Mode (Bus Error on Write)

## Signal Processing Unit 2 (SPU2)

Table 962 Reset Values of CFARCFG2

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## CFAR Configuration 3

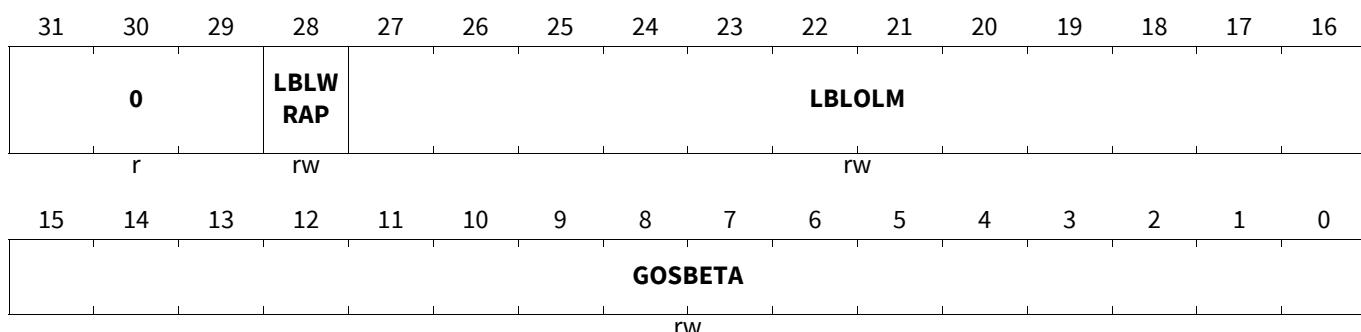
Configuration Parameters for the integrated CFAR module are stored in this register

## CFARCFG3

## CFAR Configuration 3

(002F8<sub>H</sub>)

Reset Value: Table 964



Field	Bits	Type	Description
GOSBETA	15:0	rw	<b>GOS-CFAR Beta</b> Additive constant scaling the GOS-CFAR threshold
LBLOLM	27:16	rw	<b>Label Output Limit</b> Define the maximum number of target labels to write to memory. The number of labels is the value of this field plus one so a value of 3 will result in 4 labels being written. When enabled to do so the target labeling function shall generate an interrupt when the target count reaches half the value in this field and when it reaches the full value. The half way value shall be calculated by shifting the full value right by one place. The minimum valid value of this field is 2 <sub>D</sub> when the target label output function is enabled CFARCTRL.LBLMODE==INT or NOINT. The behaviour when LBLOLM is zero or one in these cases is undefined.
LBLWRAP	28	rw	<b>Label List Wrap Mode Enable</b> If set, then when the counter LBLCNT.CNT reaches the value in LBLOLM the count shall wrap to zero and the output address shall be reset to the value in LBLBASE.BASE the next time the Logical Function equals 1 <sub>B</sub> . This bit has no effect when CFARCTRL.LBLMODE is equal to COUNT.
0	31:29	r	<b>RESERVED</b>

## Signal Processing Unit 2 (SPU2)

Table 963 Access Mode Restrictions of CFARCFG3 sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	GOSBETA, LBLLOLM, LBLWRAP	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	GOSBETA, LBLLOLM, LBLWRAP	Default Access Mode (Bus Error on Write)

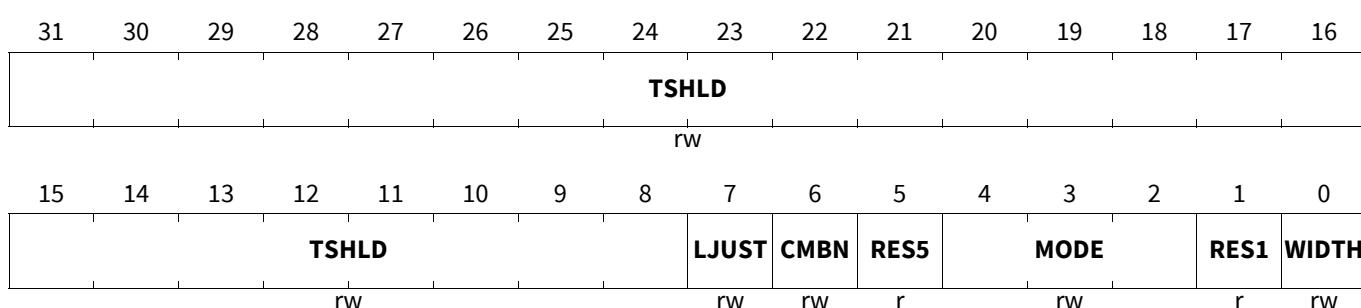
Table 964 Reset Values of CFARCFG3

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Local Maximum Control

Control Parameters for the Local Maximum Unit

## LCLMAX

Local Maximum Control (002FC<sub>H</sub>) Reset Value: Table 966

Field	Bits	Type	Description
<b>WIDTH</b>	0	rw	<b>Local Maximum Window Width</b> This bitfield sets the width of the comparison window for the Local Maximum detection. <ul style="list-style-type: none"> <li>0<sub>B</sub> <b>THREE</b>, Window Size of Three The bin under consideration will be flagged if it is a local maximum compared to the adjacent bins in the FFT result</li> <li>1<sub>B</sub> <b>FIVE</b>, Window Size of Five The bin under consideration will be flagged if it is a local maximum compared to the four adjacent bins (index -2 to index +2) in the FFT result</li> </ul>
<b>RES1</b>	1	r	<b>Reserved</b>

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
MODE	4:2	rw	<b>Operating Mode</b> Operating mode for Local Maximum engine. Controls both the threshold and local maximum functions. 000 <sub>B</sub> <b>TOFF_LUNDER</b> , No thresholding, Flag bin if not Local Maximum 001 <sub>B</sub> <b>TOFF_LOVER</b> , No thresholding, Flag bin if Local Maximum 010 <sub>B</sub> <b>TUNDER_LOFF</b> , Flag Bin if Under Threshold, No Local Maximum 011 <sub>B</sub> <b>TOVER_LOFF</b> , Flag bin if over threshold, No Local Maximum 100 <sub>B</sub> <b>TUNDER_LUNDER</b> , Flag Bin if Under Threshold, Flag bin if not Local Maximum 101 <sub>B</sub> <b>TUNDER_LOVER</b> , Flag Bin if Under Threshold, Flag bin if Local Maximum 110 <sub>B</sub> <b>TOVER_LUNDER</b> , Flag bin if over threshold, Flag bin if not Local Maximum 111 <sub>B</sub> <b>TOVER_LOVER</b> , Flag bin if over threshold, Flag bin if Local Maximum
RES5	5	r	<b>Reserved</b>
CMBN	6	rw	<b>Check Combine</b> This bitfield is used to define how the results of the two possible checks are combined. This field only has an effect if both checks are enabled. 0 <sub>B</sub> <b>OR</b> , OR Check Results Flag bin if either check triggers rejection 1 <sub>B</sub> <b>AND</b> , AND Check Results. Flag bin if both checks trigger a rejection
LJUST	7	rw	<b>Left Justify</b> The 24 bit TSHLD bitfield is compared against a 32 bit value. This field controls whether the comparison is made against the 24 MSBs or 24 LSBs 0 <sub>B</sub> <b>RIGHT</b> , Right Justify. Pad with 8 MSBs to create a 32 bit comparison value 1 <sub>B</sub> <b>LEFT</b> , Left Justify. Pad with 8 LSBs to create a 32 bit comparison value
TSHLD	31:8	rw	<b>Threshold</b> Threshold Value to be Used for Bin Magnitude Comparison

Table 965 Access Mode Restrictions of **LCLMAX** sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES1, RES5	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	CMBN, LJUST, MODE, TSHLD, WIDTH	
Otherwise (default)	r	CMBN, LJUST, MODE, RES1, RES5, TSHLD, WIDTH	

## Signal Processing Unit 2 (SPU2)

Table 966 Reset Values of LCLMAX

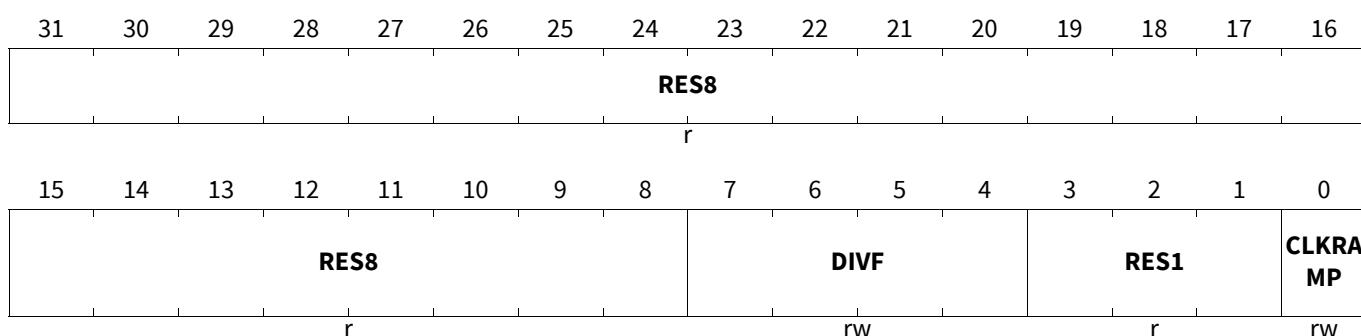
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Auxiliary Control Information

This register is used for SPU control information

## AUXCTRL

Auxiliary Control Information (00300<sub>H</sub>) Reset Value: Table 968



Field	Bits	Type	Description
CLKRAMP	0	rw	<b>Clock Ramp</b> If enabled, the contents of the first and last input buffer RAM of the configuration will be processed by the LOADER/MATH0/MATH1/UNLOADER pipeline at 50% of the SPU clock frequency. 0 <sub>B</sub> OFF, Clock Ramp is Disabled 1 <sub>B</sub> ON, Clock Ramp is Enabled
RES1	3:1	r	<b>Reserved</b>
DIVF	7:4	rw	<b>Clock Division Fraction</b> The contents of this field is an unsigned integer used to set the fractional clock ratio. If this field is non-zero, the LOADER/MATH0/MATH1/FFT/UNLOADER pipeline will be clocked at a fraction of the SPU clock with a frequency equal to the SPU clock frequency multiplied by the fraction (1 - DIVF/16).
RES8	31:8	r	<b>Reserved</b>

## Signal Processing Unit 2 (SPU2)

**Table 967 Access Mode Restrictions of AUXCTRL sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN and Write Accesses	r	RES1, RES8	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	CLKRAMP, DIVF	
Otherwise (default)	r	CLKRAMP, DIVF, RES1, RES8	Default Access Mode (Bus Error on Write)

**Table 968 Reset Values of AUXCTRL**

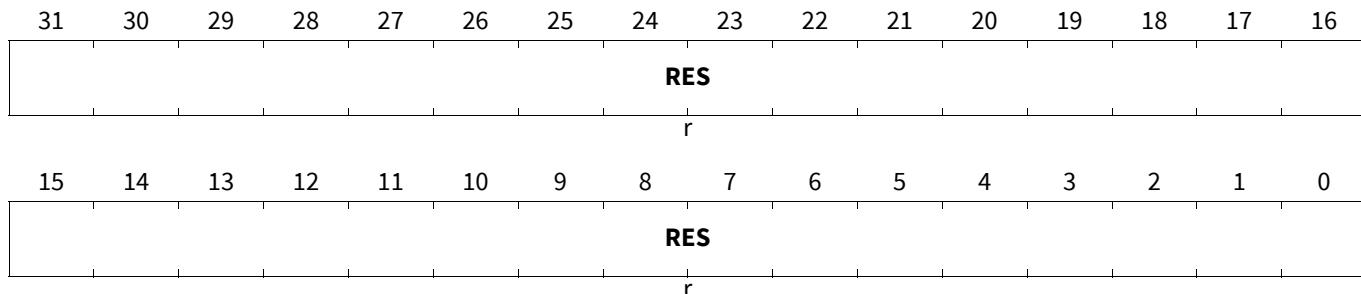
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Spare Configuration Register

This register is reserved for future expansion of the SPU functionality

### ACFG0

**Spare Configuration Register** (00304<sub>H</sub>) Reset Value: [Table 970](#)



Field	Bits	Type	Description
<b>RES</b>	31:0	r	<b>Reserved</b>

**Table 969 Access Mode Restrictions of ACFG0 sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)

## Signal Processing Unit 2 (SPU2)

**Table 970 Reset Values of ACFG0**

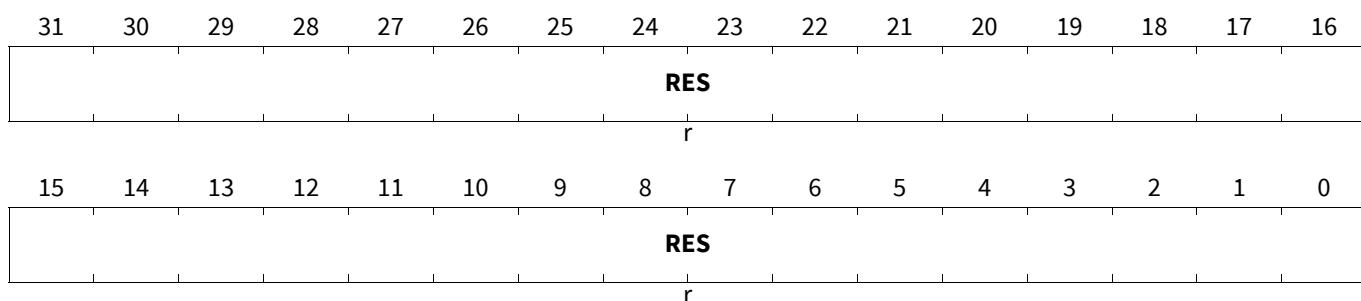
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Spare Configuration Register

This register is reserved for future expansion of the SPU functionality

**ACFG1**

**Spare Configuration Register** **Reset Value: Table 972**



Field	Bits	Type	Description
RES	31:0	r	Reserved

**Table 971 Access Mode Restrictions of ACFG1 sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)

**Table 972 Reset Values of ACFG1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

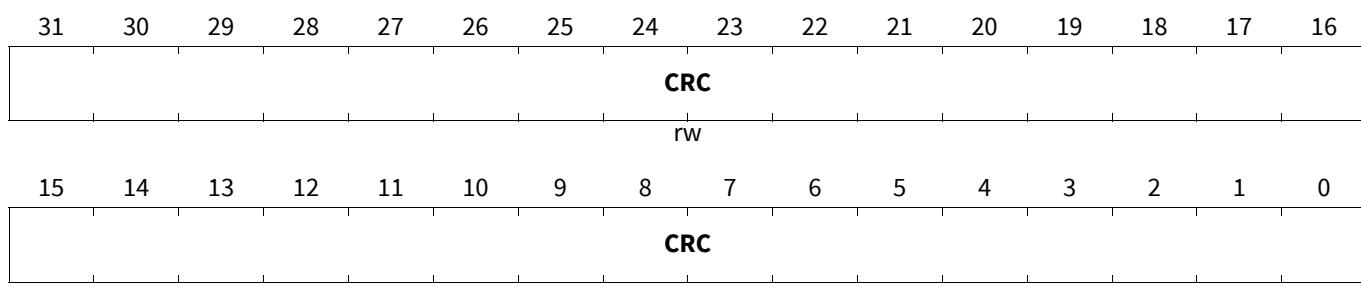
### Register CRC

Expected CRC value for all registers from ID\_CONF to CTRL (excluding the CRC register itself)

## Signal Processing Unit 2 (SPU2)

### REGCRC

#### Register CRC

(0030C<sub>H</sub>)Reset Value: [Table 974](#)

Field	Bits	Type	Description
CRC	31:0	rw	<b>CRC</b> The expected 32 bit CRC of all registers from ID_CONF to CTRL excluding the REGCRC registeritself (which is replaced by 0x00000000)

**Table 973 Access Mode Restrictions of REGCRC sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	CRC	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	CRC	Default Access Mode (Bus Error on Write)

**Table 974 Reset Values of REGCRC**

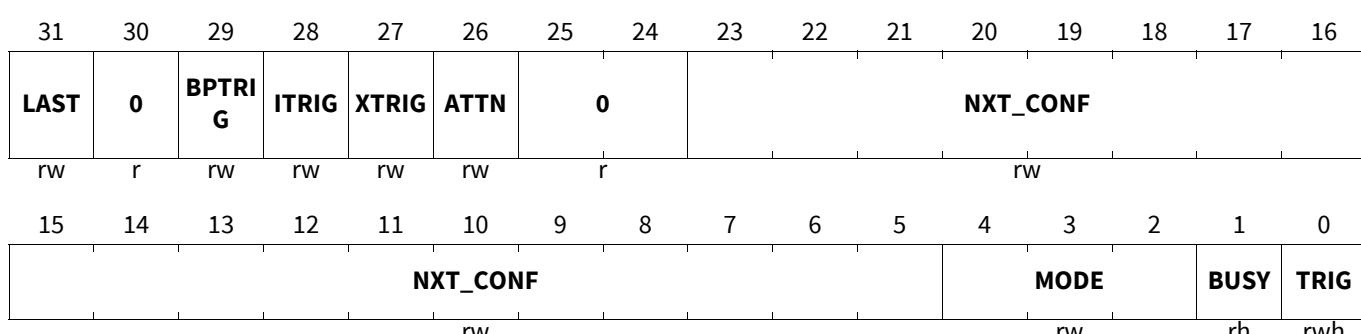
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### SPU Control

Control Register for setting the SPU mode and triggering processing operations

### CTRL

#### SPU Control

(00310<sub>H</sub>)Reset Value: [Table 976](#)

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>TRIG</b>	0	rwh	<b>SPU Software Trigger</b> Write 1 to trigger the SPU by software. Always reads as 0. $0_B$ <b>NULL</b> , No Operation $1_B$ <b>TRIGGER</b> , Trigger the SPU to commence data processing
<b>BUSY</b>	1	rh	<b>SPU Busy Flag</b> $0_B$ <b>DONE</b> , SPU Idle SPU has finished processing and is guaranteed not to be using EMEM $1_B$ <b>BUSY</b> , SPU Busy SPU is currently processing data
<b>MODE</b>	4:2	rw	<b>SPU Trigger Mode</b> $000_B$ <b>OFF</b> , SPU is disabled $001_B$ <b>INT</b> , Internal Trigger $010_B$ <b>EXT</b> , External Trigger $011_B$ <b>SPU0</b> , Trigger on SPU0 Done $100_B$ <b>SPU1</b> , Trigger on SPU1 Done $101_B$ <b>RELOAD</b> , Reload configuration from memory with no processing Although the SPU main functions shall not be triggered in this mode the Bypass function shall be triggered if BPTRIG bit is set. $110_B$ <b>SW</b> , Software Trigger $111_B$ <b>STOP</b> , SPU will stop at end of current operation.
<b>NXT_CONF</b>	23:5	rw	<b>Next Configuration Base Address</b> This is the base address in the configuration RAM of the next set of configuration settings for the SPU. This is a byte address offset into the configuration RAM (i.e. first location in the configuration RAM is address 0) but it must be 64 bit aligned (i.e. bits [2:0] of the address must be 0).
<b>ATTN</b>	26	rw	<b>Generate Service Request Interrupt</b> If this bit is set in a loaded configuration and the related mask bit, STATCTRL.INTMSK(0), is also set, then a service request interrupt will be generated when the execution of the configuration is completed, even if this is not the last configuration in a linked list. $0_B$ <b>OFF</b> , No effect $1_B$ <b>ON</b> , A service request interrupt will be generated when execution of the loaded configuration has completed
<b>XTRIG</b>	27	rw	<b>Cross Trigger</b> If this bit is set a "DONE" event will be triggered at the end of execution of the current configuration (specifically when the last data has been written to Radar memory). This will be used by the SPU0 or SPU1 trigger modes defined for the CTRL.MODE bitfield

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>ITRIG</b>	28	rw	<b>Incremental Trigger</b> Control of Incremental Trigger Output. If enabled, the SPU will generate a pulse on the incremental trigger output after the MATH2 unit has completed processing the contents of the buffer memory and the ODM has successfully written all data to memory 0 <sub>B</sub> <b>OFF</b> , Incremental Trigger is Disabled 1 <sub>B</sub> <b>ON</b> , Incremental Trigger is Enabled
<b>BPTRIG</b>	29	rw	<b>Bypass Trigger</b> If this bit is set the Bypass function shall be triggered when the trigger condition specified by the MODE field occurs. The MODE==RELOAD may be used to trigger Bypass without triggering normal SPU operation.
<b>LAST</b>	31	rw	<b>Last Configuration</b> If this bit is set while loading a configuration, this will be the last configuration processed and no further configurations will be loaded
<b>0</b>	25:24, 30	r	<b>Reserved</b> Reserved. This was the DIV field which has been replaced by the fractional clock divider control.

**Table 975 Access Mode Restrictions of CTRL sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rh	BUSY	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	ATTN, BPTRIG, LAST, MODE, NXT_CONF, XTRIG	
	wh	TRIG	
Otherwise (default)	r	ATTN, BPTRIG, LAST, MODE, NXT_CONF, XTRIG	Default Access Mode (Bus Error on Write)
	rh	BUSY, TRIG	

**Table 976 Reset Values of CTRL**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Burst Count

The number of interference bursts detected by the detector in the MATH0 unit.

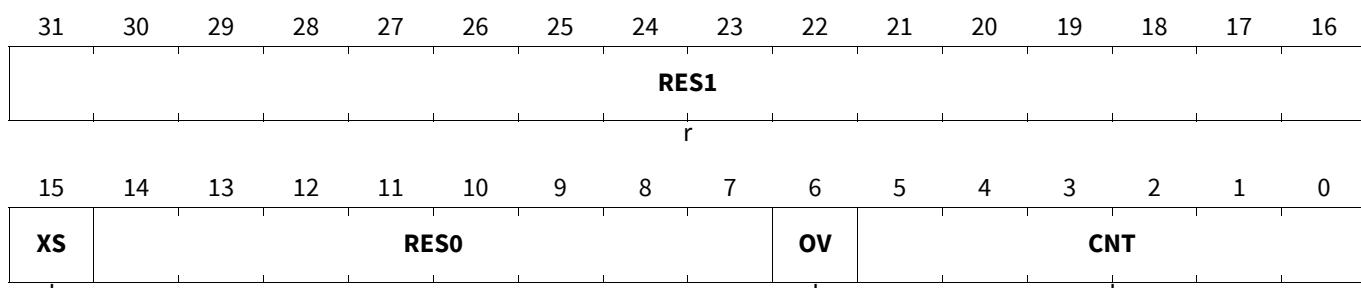
## Signal Processing Unit 2 (SPU2)

### MDM0\_CNT

#### Burst Count

(00314<sub>H</sub>)

Reset Value: [Table 978](#)



Field	Bits	Type	Description
<b>CNT</b>	5:0	rh	<b>Burst Count</b> Counter incremented every time an interference burst is detected until the count reaches 63 <sub>D</sub> . If there are more than 63 <sub>D</sub> bursts the OV field shall be set. Reset to zero when the SPU is triggered.
<b>OV</b>	6	rh	<b>Overflow: set if burst count exceeds 16</b>
<b>RES0</b>	14:7	r	<b>Reserved: always reads as zero</b>
<b>XS</b>	15	rh	<b>Inconsistent burst data has been seen by CST unit</b> For example this will be the case when more than 16 interference bursts are detected in one ramp
<b>RES1</b>	31:16	r	<b>Reserved: always reads as zero</b>

**Table 977 Access Mode Restrictions of MDM0\_CNT sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	rh	CNT, OV, XS	Any write access will cause bus error
Otherwise (default)	rh	CNT, OV, XS	Default Access Mode (Bus Error on Write)

**Table 978 Reset Values of MDM0\_CNT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Burst descriptor

Burst descriptor register giving the location and length of a burst. The location is given by the ramp number and sample number. The first ramp and first sample are both numbered zero.

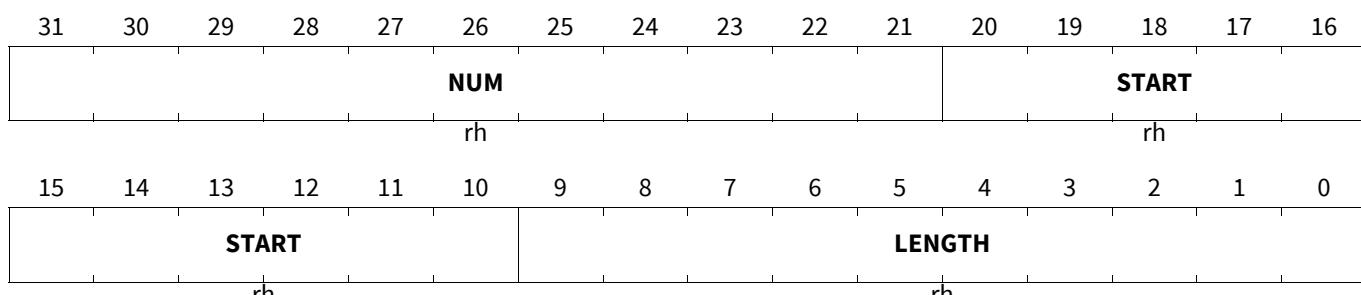
## Signal Processing Unit 2 (SPU2)

### MDM0\_BURSTb (b=0-15)

Burst descriptor

( $00318_{\text{H}} + b * 4$ )

Reset Value: [Table 980](#)



Field	Bits	Type	Description
LENGTH	9:0	rh	Length of Burst
START	20:10	rh	Start sample of the Burst
NUM	31:21	rh	Ramp Number of the Burst

**Table 979 Access Mode Restrictions of MDM0\_BURSTb (b=0-15) sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	rh	LENGTH, NUM, START	Any write access will cause bus error
Otherwise (default)	rh	LENGTH, NUM, START	Default Access Mode (Bus Error on Write)

**Table 980 Reset Values of MDM0\_BURSTb (b=0-15)**

Reset Type	Reset Value	Note
Application Reset	$0000\ 0000_{\text{H}}$	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	$0000\ 0000_{\text{H}}$	Kernel Reset (software controlled by KRST0-1 registers)

### Label List Counter

This register is duplicated so that the stored information is available for each set of configuration options used in "Double Pass" mode.

This register has a field to hold the count of the number of times the Label List has output a label. It also has a control bit to allow the count to be cleared. The count should only be cleared when the SPU is idle.

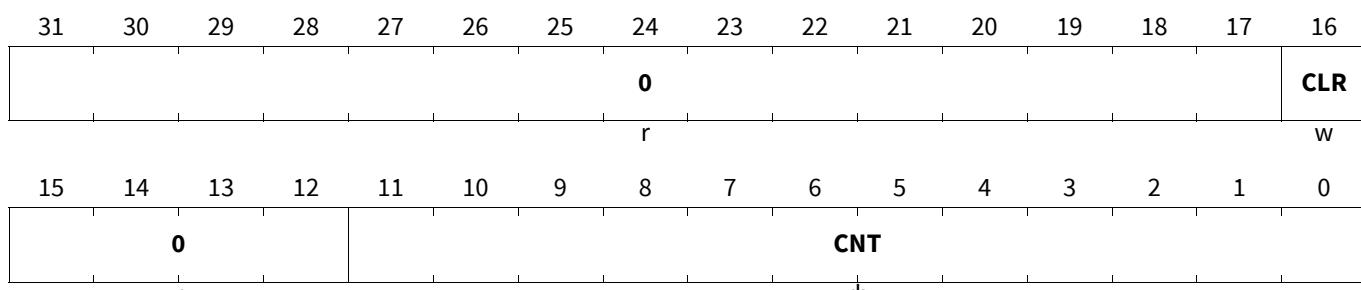
## Signal Processing Unit 2 (SPU2)

### MDq\_LBLCNT (q=0-1)

Label List Counter

(00358<sub>H</sub>+q\*54<sub>H</sub>)

Reset Value: [Table 982](#)



Field	Bits	Type	Description
CNT	11:0	rh	<b>Count</b> The count, since reset or cleared, of the number of times the Logical Function output was one.
CLR	16	w	<b>Counter clear</b> Writing a 1 <sub>B</sub> to this bit will clear the counter. Always reads as zero.
0	15:12, 31:17	r	<b>Reserved</b> Reads as 0, shall be written with 0.

**Table 981 Access Mode Restrictions of MDq\_LBLCNT (q=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses and Master enabled in ACCEN	rh	CNT	Any write access will cause bus error
Master enabled in ACCEN	w	CLR	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	rX	CLR	Default Access Mode (Bus Error on Write)
	rh	CNT	

**Table 982 Reset Values of MDq\_LBLCNT (q=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Dataset Metadata

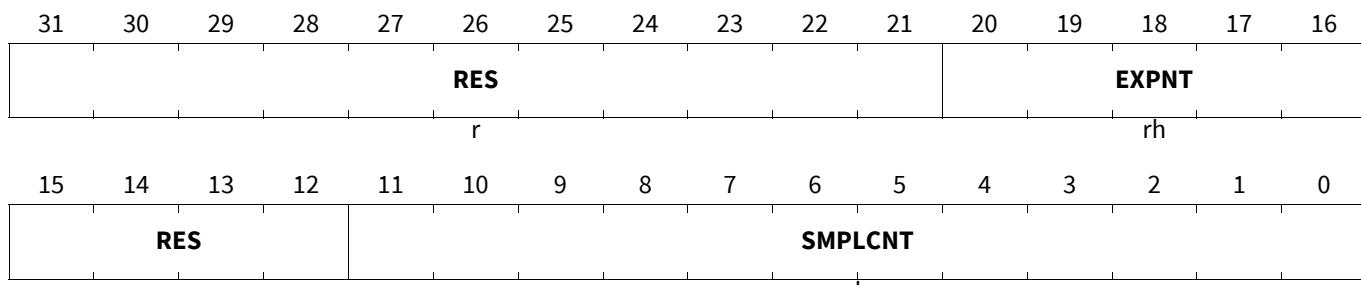
This register is duplicated so that the stored information is available for each set of configuration options used in "Double Pass" mode.

Secondary Information describing the data that has just been processed

## Signal Processing Unit 2 (SPU2)

### MDq\_METADATA (q=0-1)

Dataset Metadata

(0035C<sub>H</sub>+q\*54<sub>H</sub>)Reset Value: [Table 984](#)

Field	Bits	Type	Description
<b>SMPLCNT</b>	11:0	rh	<b>Sample Count</b> Number of samples per FFT written to memory. This is useful when using the CFAR inline as the number of bins in the FFT result that will pass the threshold test is not known ahead of processing. The field contains the actual number of samples written to the Radar Memory
<b>RES</b>	15:12, 31:21	r	<b>Reserved</b>
<b>EXPNT</b>	20:16	rh	<b>Common Exponent</b> This indicates the optimum alignment correction that could have been applied if reformatting to 16 bit precision. If "n" is the minimum number of redundant sign bits present in the data at the FFT output, it will be set to $(16_D - n)$ . This can then be used as the value of UNLDR_CONF.EXPNT to be programmed for the next measurement cycle if it is intended to use 16 bit precision data. The maximum permissible value for this field is $16_D$

**Table 983 Access Mode Restrictions of MDq\_METADATA (q=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
	rh	EXPNT, SMPLCNT	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	EXPNT, SMPLCNT	

**Table 984 Reset Values of MDq\_METADATA (q=0-1)**

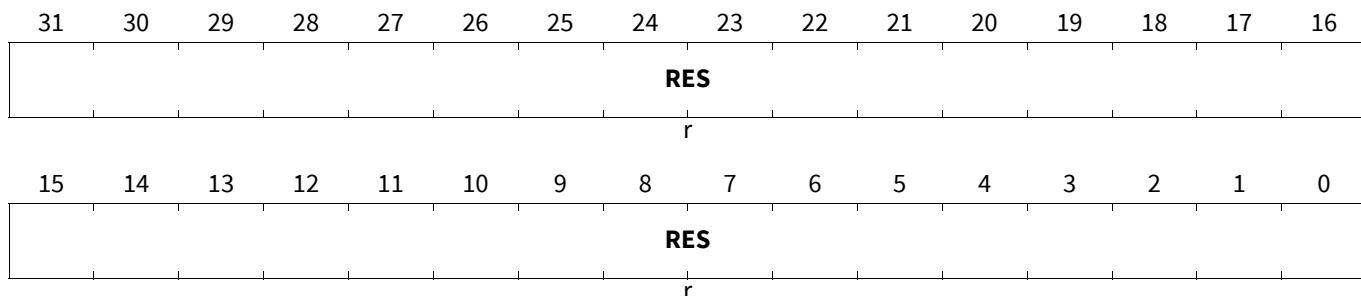
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit 2 (SPU2)

### Reserved

#### MDq\_SPAREm (m=0-18;q=0-1)

Reserved

(00360<sub>H</sub>+q\*54<sub>H</sub>+m\*4)Reset Value: [Table 986](#)

Field	Bits	Type	Description
RES	31:0	r	Reserved

**Table 985 Access Mode Restrictions of MDq\_SPAREm (m=0-18;q=0-1) sorted by descending priority**

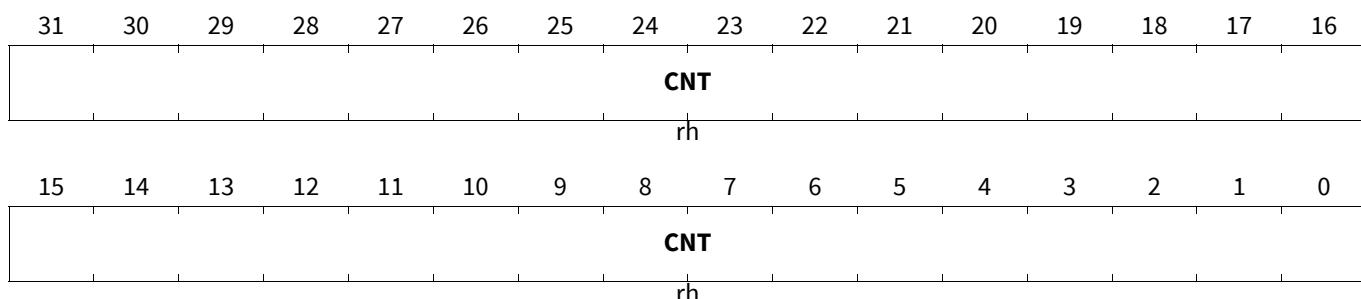
Mode Name	Access Mode		Description
Write Accesses	r	RES	Any write access will cause bus error
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)

**Table 986 Reset Values of MDq\_SPAREm (m=0-18;q=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### IDLCNT

#### Idle Count

(00400<sub>H</sub>)Reset Value: [Table 988](#)

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
CNT	31:0	rh	Idle Count

**Table 987 Access Mode Restrictions of IDLCNT sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	rh	CNT	Any write access will cause bus error
Otherwise (default)	rh	CNT	Default Access Mode (Bus Error on Write)

**Table 988 Reset Values of IDLCNT**

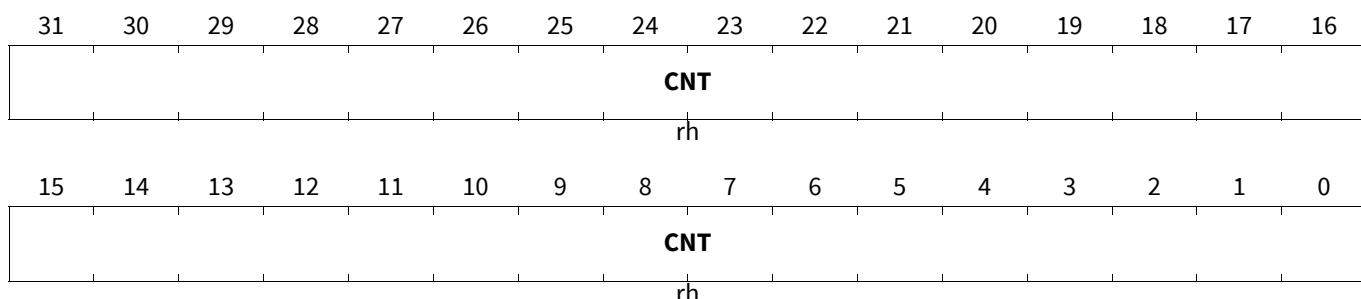
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Execution Count

Number of clock cycles taken to execute a configuration. Reset to zero when the SPU is triggered. Incremented each clock cycle when SPU is running. Stops when CTRL.BUSY transitions to DONE.

#### EXCNT

**Execution Count** Reset Value: [Table 990](#)



Field	Bits	Type	Description
CNT	31:0	rh	Count

**Table 989 Access Mode Restrictions of EXCNT sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	rh	CNT	Any write access will cause bus error
Otherwise (default)	rh	CNT	Default Access Mode (Bus Error on Write)

## Signal Processing Unit 2 (SPU2)

**Table 990 Reset Values of EXCNT**

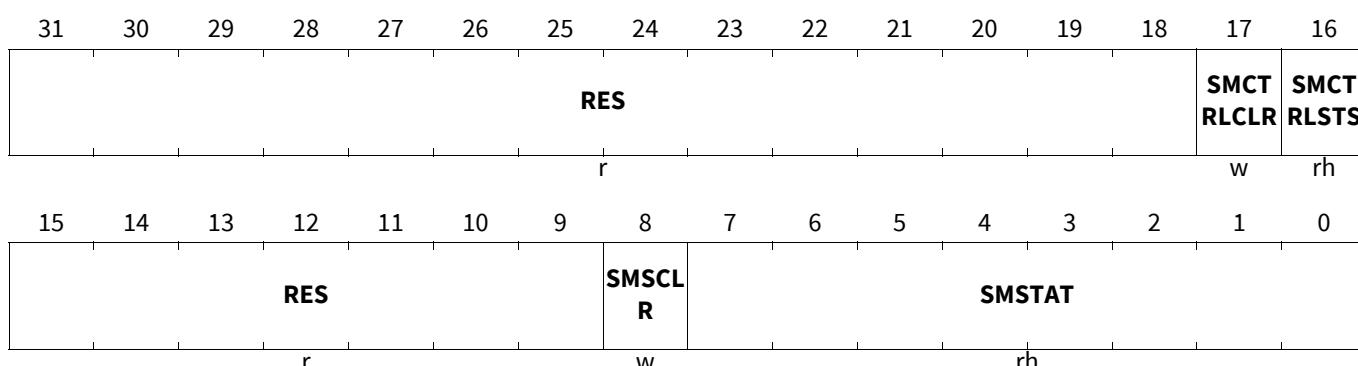
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Safety Mechanism Status

The SPU contains hardware safety mechanisms which monitor key areas of the SPU for correct functioning. In the event of an alarm being triggered by one of the safety mechanisms, this register can be read to determine which of the safety mechanisms has detected an error

#### SMSTAT

**Safety Mechanism Status** (00408<sub>H</sub>) **Reset Value:** [Table 992](#)



Field	Bits	Type	Description
<b>SMSTAT</b>	7:0	rh	<p><b>Safety Mechanism Status</b></p> <p>Each bit will be set to one if the associated safety mechanism has detected a fault. The flags can be cleared by writing 1<sub>b</sub> to SMSTAT.SMSCLR. The values read from this bitfield are interpreted as follows (multiple failure conditions will cause the individual values to be summed):</p> <ul style="list-style-type: none"> <li>01<sub>H</sub> <b>RIFCRC</b>, Check of computed CRC for RIF data against received value has failed</li> <li>02<sub>H</sub> <b>REGCRC</b>, Check of computed Register CRC against register value has failed</li> <li>04<sub>H</sub> <b>BYPASSCRC</b>, Check of computed Bypass Data CRC against value read from radar memory has failed</li> <li>08<sub>H</sub> <b>RDECC</b>, ECC Check Fail on Read from Radar Memory</li> <li>10<sub>H</sub> <b>RMCTRL</b>, Radar Memory Access Control Signals failed consistency check</li> <li>20<sub>H</sub> <b>RMTAERR</b>, Access to Radar Memory has resulted in an Error</li> <li>40<sub>H</sub> <b>R2BPCRC</b>, Check of computed CRC for RIF data in the Bypass module against received value has failed</li> <li>80<sub>H</sub> <b>FFTECC</b>, ECC check fail in register-based FFT memories</li> </ul>

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>SMSCLR</b>	8	w	<b>Clear Safety Mechanism Status</b> Write 1 <sub>b</sub> to this bit to clear the safety mechanism status flags. This bit will always read 0 <sub>b</sub> .
<b>RES</b>	15:9, 31:18	r	<b>Reserved</b>
<b>SMCTRLSTS</b>	16	rh	<b>Safety Mechanism Control Status</b> Safety Mechanism Control Logic Check has failed (inconsistent logic state)
<b>SMCTRLCLR</b>	17	w	<b>Clear SMCTRL Status Flag</b> Write 1 <sub>b</sub> to this bit to clear the SMCTRL status flag (SMSTAT.SMCTRLSTS). This bit will always read 0 <sub>b</sub> .

**Table 991 Access Mode Restrictions of SMSTAT sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rh	SMCTRLSTS, SMSTAT	
	w	SMCTRLCLR, SMSCLR	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rX	SMCTRLCLR, SMSCLR	
	rh	SMCTRLSTS, SMSTAT	

**Table 992 Reset Values of SMSTAT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Safety Mechanism Control Functions

This register provides access to the control functions for safety mechanisms which are not normally expected to be modified as part of the normal configuration of the SPU.

#### SMCTRL

#### Safety Mechanism Control Functions

(0040C<sub>H</sub>)

Reset Value: [Table 994](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FFTECC	R2BPCRC	RMTAERR	BPCRC	RIFCRC	REGCRCEN	DATACRCEN	CTRLCRCEN								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>CTRLCRCEN</b>	1:0	rw	<p><b>Control CRC Enable</b></p> <p>This bitfield allows control of the CRC blocks monitoring data invariant signals. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set</li> <li>01<sub>B</sub> <b>OFF</b>, CRC Monitors are disabled</li> <li>10<sub>B</sub> <b>ON</b>, CRC Monitors are enabled</li> <li>11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</li> </ul>
<b>DATA_CRCEN</b>	3:2	rw	<p><b>Data CRC Enable</b></p> <p>This bitfield allows control of the CRC blocks monitoring data dependent signals. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set</li> <li>01<sub>B</sub> <b>OFF</b>, CRC Monitors are disabled</li> <li>10<sub>B</sub> <b>ON</b>, CRC Monitors are enabled</li> <li>11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</li> </ul>
<b>REGCRCEN</b>	5:4	rw	<p><b>Register CRC Enable</b></p> <p>Enable a Periodic CRC check of the register values against a known CRC stored in REGCRC.CRC. This checks registers at addresses between ID_CONF and CTRL inclusive but excluding the REGCRC register. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set</li> <li>01<sub>B</sub> <b>OFF</b>, CRC is disabled</li> <li>10<sub>B</sub> <b>ON</b>, CRC is enabled</li> <li>11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</li> </ul>
<b>RIFCRC</b>	7:6	rw	<p><b>RIF Data CRC Check Enable</b></p> <p>Enable a CRC Check on the incoming RIF data. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set</li> <li>01<sub>B</sub> <b>OFF</b>, CRC is disabled</li> <li>10<sub>B</sub> <b>ON</b>, CRC is enabled</li> <li>11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</li> </ul>
<b>BPCRC</b>	9:8	rw	<p><b>Bypass Data CRC Check Enable</b></p> <p>Enable a CRC Check on the bypassed RIF data. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set</li> <li>01<sub>B</sub> <b>OFF</b>, CRC is disabled</li> <li>10<sub>B</sub> <b>ON</b>, CRC is enabled</li> <li>11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</li> </ul>
<b>RMTAERR</b>	11:10	rw	<p><b>Radar Memory Access Address Error Enable</b></p> <p>Enable an alarm to be generated if an access to the Radar Memory returns an error due to an invalid address. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set</li> <li>01<sub>B</sub> <b>OFF</b>, Access Address Error is disabled</li> <li>10<sub>B</sub> <b>ON</b>, Access Address Error is enabled</li> <li>11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</li> </ul>

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
R2BPCRC	13:12	rw	<b>Incoming RIF Data Bypass module CRC Check Enable</b> Enable a CRC Check on the incoming RIF data in the Bypass module. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>OFF</b> , CRC is disabled 10 <sub>B</sub> <b>ON</b> , CRC is enabled 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set
FFTECC	15:14	rw	<b>FFT Register-Based Memory ECC Check Enable</b> Enable ECC check on register-based FFT memories. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>OFF</b> , ECC is disabled 10 <sub>B</sub> <b>ON</b> , ECC is enabled 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set
RES	31:16	r	<b>Reserved</b>

**Table 993 Access Mode Restrictions of SMCTRL sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BPCRC, CTRLCRCEN, DATACRCEN, FFTECC, R2BPCRC, REGCRCEN, RIFCRC, RMTAERR	
Otherwise (default)	r	BPCRC, CTRLCRCEN, DATACRCEN, FFTECC, R2BPCRC, REGCRCEN, RES, RIFCRC, RMTAERR	Default Access Mode (Bus Error on Write)

**Table 994 Reset Values of SMCTRL**

Reset Type	Reset Value	Note
Application Reset	0000 5555 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 5555 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Signal Processing Unit 2 (SPU2)

### SPU Monitor

#### MONITOR

##### SPU Monitor

(00410<sub>H</sub>)

Reset Value: [Table 996](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BP_BU SY	M0_B USY	ODM_ BUSY	M2_B USY	UL_BU SY	M1_B USY	LDR_B USY	IDM_B USY	0							SAMPLE
rh	rh	rh	rh	rh	rh	rh	rh	r							rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															RAMP

rh                    r                    rh

Field	Bits	Type	Description
RAMP	10:0	rh	<b>Ramp Counter</b> Ramp Count for current measurement cycle. Only active when data is being loaded from the RIF
SAMPLE	22:12	rh	<b>Sample Count</b> Sample count for current ramp. Only active when data is being loaded from the RIF
IDM_BUSY	24	rh	<b>IDM Busy</b>
LDR_BUSY	25	rh	<b>Loader Busy</b>
M1_BUSY	26	rh	<b>MATH1 Unit Busy</b>
UL_BUSY	27	rh	<b>Unloader Busy</b>
M2_BUSY	28	rh	<b>MATH2 Busy</b>
ODM_BUSY	29	rh	<b>ODM Busy</b>
M0_BUSY	30	rh	<b>MATH0 Unit Busy</b>
BP_BUSY	31	rh	<b>Bypass Function Busy</b>
0	11, 23	r	<b>Reserved</b>

**Table 995 Access Mode Restrictions of MONITOR sorted by descending priority**

Mode Name	Access Mode		Description
Otherwise and Write Accesses	rh	BP_BUSY, IDM_BUSY, LDR_BUSY, M0_BUSY, M1_BUSY, M2_BUSY, ODM_BUSY, RAMP, SAMPLE, UL_BUSY	Read Only
Otherwise (default)	rh	BP_BUSY, IDM_BUSY, LDR_BUSY, M0_BUSY, M1_BUSY, M2_BUSY, ODM_BUSY, RAMP, SAMPLE, UL_BUSY	Default Access Mode (Bus Error on Write)

## Signal Processing Unit 2 (SPU2)

**Table 996 Reset Values of MONITOR**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Safety Mechanism Control Functions (User)

This register provides access to the control functions for safety mechanisms not used as part of the normal configuration of the SPU. The various test bitfields defined in this register and used to test that the safety mechanisms can generate alarms are intended as a connectivity test only. It is therefore not necessary (or desirable) for the SPU to be running while these tests are run.

#### SMUSER

#### Safety Mechanism Control Functions [User] (00414<sub>H</sub>)

Reset Value: [Table 998](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>RES</b>	<b>FFTECCTST</b>	<b>R2BPCRCTST</b>	<b>RMTAERRTST</b>	<b>RDECCTST</b>	<b>RMCTRLTST</b>	<b>BPCRCTST</b>	<b>RIFCRCTST</b>								
r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>RES</b>															<b>CINIT</b>
r															rw

Field	Bits	Type	Description
<b>CINIT</b>	1:0	rw	<b>Monitor CRC Unit Initialise</b> Write 10 <sub>b</sub> to initialise the Monitor CRC Units. The CRC monitors will be held in the initialisation state until 01 <sub>b</sub> is written. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit, 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>NOP</b> , No Effect 10 <sub>B</sub> <b>ON</b> , Monitor CRC Units Initialised 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set
<b>RES</b>	15:2, 31:30	r	<b>Reserved</b>
<b>RIFCRCTST</b>	17:16	rw	<b>Test RIF CRC</b> Inject a deliberate error into the mechanism checking the CRC of RIF data to test the alarm generation. When set to 10 <sub>b</sub> , all RIF CRC checks should fail until this field is written with 01 <sub>b</sub> . Writing an invalid value will set the SMSTAT.SMCTRLSTS bit. 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>OFF</b> , No Error Injected 10 <sub>B</sub> <b>ON</b> , Error Injected 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
<b>BPCRCTST</b>	19:18	rw	<p><b>Test Bypass CRC</b></p> <p>Inject a deliberate error into the mechanism checking the CRC of bypass the data to test the alarm generation. When set to 10b, all RIF CRC checks should fail until this field is written with 01b. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit.</p> <p>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set            01<sub>B</sub> <b>OFF</b>, No Error Injected            10<sub>B</sub> <b>ON</b>, Error Injected            11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>
<b>RMCTRLTST</b>	21:20	rw	<p><b>Test Radar Memory Control</b></p> <p>Inject a deliberate error into the mechanism checking the consistency of the Radar Memory control signals to test the alarm generation. When set to 10<sub>b</sub>, an alarm will be flagged until this field is written with 01<sub>b</sub>. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit.</p> <p>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set            01<sub>B</sub> <b>OFF</b>, No Error Injected            10<sub>B</sub> <b>ON</b>, Error Injected            11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>
<b>RDECCTST</b>	23:22	rw	<p><b>Test EMEM Read Data ECC</b></p> <p>Inject a deliberate error into the mechanism checking the ECC of data read from the Radar Memory to test the alarm generation. When set to 10b, all ECC checks should fail until this field is written with 01b. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit.</p> <p>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set            01<sub>B</sub> <b>OFF</b>, No Error Injected            10<sub>B</sub> <b>ON</b>, Error Injected            11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>
<b>RMTAERRTST</b>	25:24	rw	<p><b>Test Radar Memory Access Address Error</b></p> <p>Inject a deliberate error into the mechanism checking for an address error condition being flagged for a access to Radar Memory to test the alarm generation. When set to 10b, all SPU accesses to Radar Memory should result in a alarm until this field is written with 01b. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit.</p> <p>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set            01<sub>B</sub> <b>OFF</b>, No Error Injected            10<sub>B</sub> <b>ON</b>, Error Injected            11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>
<b>R2BPCRCTST</b>	27:26	rw	<p><b>Test Incoming RIF Data Bypass module CRC Check</b></p> <p>Inject a deliberate error into the mechanism checking the CRC of RIF data in the Bypass module to test the alarm generation. When set to 10b, all RIF Bypass CRC checks should fail until this field is written with 01b. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit.</p> <p>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set            01<sub>B</sub> <b>OFF</b>, No Error Injected            10<sub>B</sub> <b>ON</b>, Error Injected            11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
FFTECCTST	29:28	rw	<p><b>Test FFT Register-based Memory ECC</b></p> <p>Inject a deliberate error into the mechanism checking the ECC of register-based FFT memory to test the alarm generation. When set to 10b, all ECC checks should fail until this field is written with 01b. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit.</p> <p>00<sub>B</sub> <b>ERR0</b>, Invalid. SMCTRLSTS will be set</p> <p>01<sub>B</sub> <b>OFF</b>, No Error Injected</p> <p>10<sub>B</sub> <b>ON</b>, Error Injected</p> <p>11<sub>B</sub> <b>ERR3</b>, Invalid. SMCTRLSTS will be set</p>

**Table 997 Access Mode Restrictions of SMUSER sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	BPCRCTST, CINIT, FFTECCTST, R2BPCRCTST, RDECCTST, RIFCRCTST, RMCTRLTST, RMTAERRTST	
Otherwise (default)	r	BPCRCTST, CINIT, FFTECCTST, R2BPCRCTST, RDECCTST, RES, RIFCRCTST, RMCTRLTST, RMTAERRTST	Default Access Mode (Bus Error on Write)

**Table 998 Reset Values of SMUSER**

Reset Type	Reset Value	Note
Application Reset	1555 0001 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	1555 0001 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Monitor CRC Register

CRC monitor data dependent signals within the SPU.

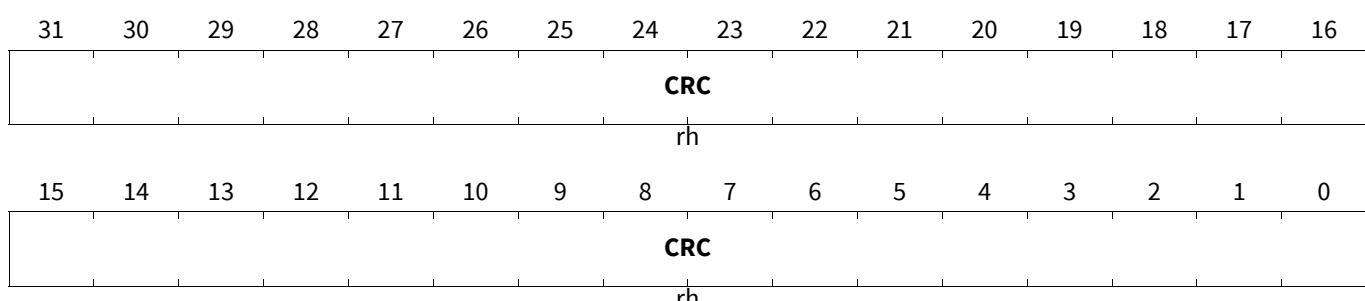
Allows read access to the output from one of the Monitor CRC Units

#### DATAd\_CRC (d=0-85)

#### Monitor CRC Register

(00418<sub>H</sub>+d\*4)

Reset Value: [Table 1000](#)



## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
CRC	31:0	rh	<b>CRC</b> CRC value. Write 10 <sub>b</sub> to SMUSER.CINIT to clear

**Table 999 Access Mode Restrictions of **DATAd\_CRC (d=0-85)** sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	rh	CRC	Any write access will cause bus error
Otherwise (default)	rh	CRC	Default Access Mode (Bus Error on Write)

**Table 1000 Reset Values of **DATAd\_CRC (d=0-85)****

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### CRC of Data Monitor CRC Registers

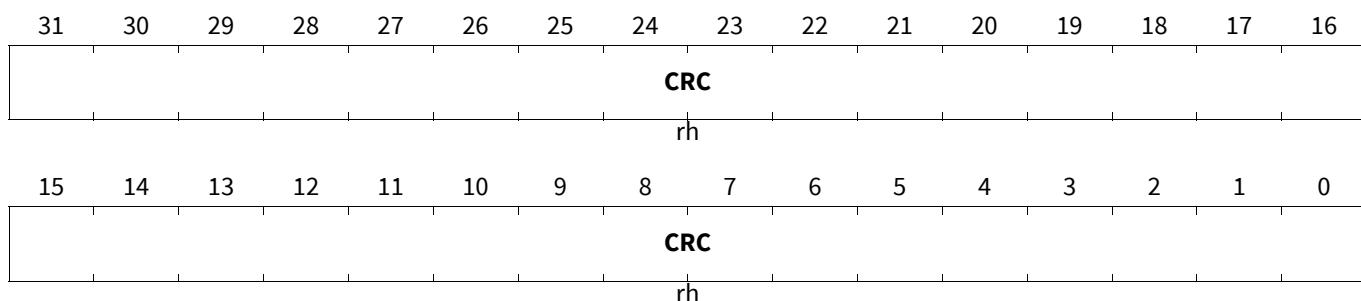
Allows read access to the value of the overall CRC of the data CRC registers.

#### CRC\_DATA\_CRC

##### CRC of Data Monitor CRC Registers

(00570<sub>H</sub>)

Reset Value: [Table 1002](#)



Field	Bits	Type	Description
CRC	31:0	rh	<b>CRC</b> CRC value. Write 10 <sub>b</sub> to SMUSER.CINIT to clear

**Table 1001 Access Mode Restrictions of **CRC\_DATA\_CRC** sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	rh	CRC	Any write access will cause bus error
Otherwise (default)	rh	CRC	Default Access Mode (Bus Error on Write)

## Signal Processing Unit 2 (SPU2)

**Table 1002 Reset Values of CRC\_DATA\_CRC**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

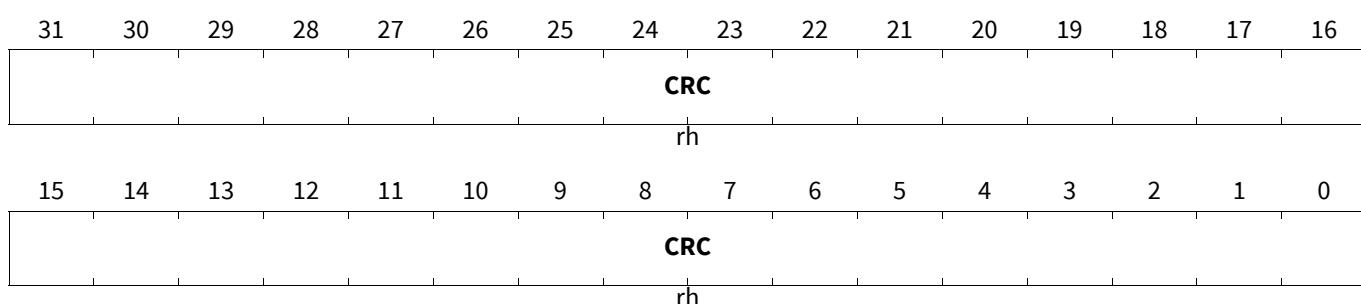
### Monitor CRC Register

CRC Monitor for data independent control signals within the SPU.

Allows read access to the output from one of the Monitor CRC Units

**CTRLe\_CRC (e=0-24)**

**Monitor CRC Register** **(00590<sub>H</sub>+e\*4)** **Reset Value: Table 1004**



Field	Bits	Type	Description
CRC	31:0	rh	<b>CRC</b> CRC value. Write 10 <sub>b</sub> to SMUSER.CINIT to clear

**Table 1003 Access Mode Restrictions of CTRLe\_CRC (e=0-24) sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	rh	CRC	Any write access will cause bus error
Otherwise (default)	rh	CRC	Default Access Mode (Bus Error on Write)

**Table 1004 Reset Values of CTRLe\_CRC (e=0-24)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### CRC of Control Monitor CRC Registers

Allows read access to the value of the overall CRC of the control CRC registers.

## Signal Processing Unit 2 (SPU2)

### CRC\_CTRL\_CRC

#### CRC of Control Monitor CRC Registers

(005F4<sub>H</sub>)

Reset Value: [Table 1006](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRC															
rh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC															
rh															

Field	Bits	Type	Description
CRC	31:0	rh	<b>CRC</b> CRC value. Write 10 <sub>b</sub> to SMUSER.CINIT to clear

**Table 1005 Access Mode Restrictions of [CRC\\_CTRL\\_CRC](#) sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	rh	CRC	Any write access will cause bus error
Otherwise (default)	rh	CRC	Default Access Mode (Bus Error on Write)

**Table 1006 Reset Values of [CRC\\_CTRL\\_CRC](#)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Control CRC Mask Register

Include Control CRC registers in the overall CRC check.

### CRC\_MASK0

#### Control CRC Mask Register

(005F8<sub>H</sub>)

Reset Value: [Table 1008](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INC31	INC30	INC29	INC28	INC27	INC26	INC25	INC24	INC23	INC22	INC21	INC20	INC19	INC18	INC17	INC16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INC15	INC14	INC13	INC12	INC11	INC10	INC9	INC8	INC7	INC6	INC5	INC4	INC3	INC2	INC1	INC0
rw															

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
INCI (i=0-31)	i	rw	<b>Include Control CRC Number i</b> Set this bit to include the corresponding Control CRC in an overall CRC calculation.

**Table 1007 Access Mode Restrictions of CRC\_MASK0 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	INCI (i=0-31)	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	INCI (i=0-31)	Default Access Mode (Bus Error on Write)

**Table 1008 Reset Values of CRC\_MASK0**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Spare Control CRC Mask Register

This register is reserved for future expansion of the SPU functionality

#### CRC\_MASK1

**Spare Control CRC Mask Register (005FC<sub>H</sub>) Reset Value: Table 1010**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								0							
									r						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								0							
									r						

Field	Bits	Type	Description
0	31:0	r	Reserved

**Table 1009 Access Mode Restrictions of CRC\_MASK1 sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	-	See bit field definitions above	Any write access will cause bus error
Otherwise (default)	-	See bit field definitions above	Default Access Mode (Bus Error on Write)

## Signal Processing Unit 2 (SPU2)

**Table 1010 Reset Values of CRC\_MASK1**

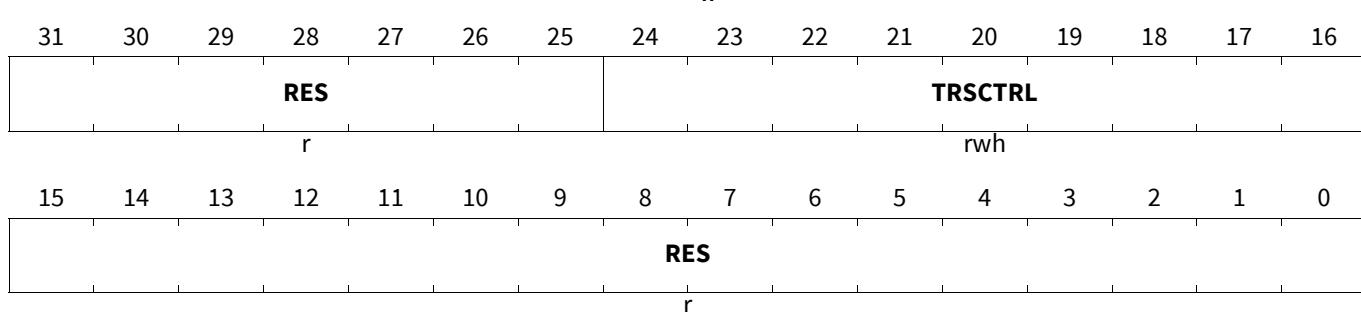
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### User OCDS Trace Control

The USROTC control register bits are only effective while the system is in debug mode. Write access requires Supervisor Mode.

#### USROTC

**User OCDS Trace Control** **Reset Value: Table 1012**



Field	Bits	Type	Description
<b>RES</b>	15:0, 31:25	r	<b>Reserved</b>
<b>TRSCTRL</b>	24:16	rwh	<b>Trace Control</b> This field individually controls the tracing of the nine possible output streams to the OCDS trace port. The nine bits in the field correspond to the nine Output DMA channels in ascending order. Any four of the streams can be simultaneously enabled for trace provided that the SPU and SRI are set to the same clock frequency. This field will revert to the reset value when OCDS is disabled.

**Table 1011 Access Mode Restrictions of USROTC sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN and Supervisor Mode	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rwh	TRSCTRL	
Supervisor Mode	r	RES	Read Access Permitted for Supervisor Mode Accesses
	rh	TRSCTRL	
Otherwise (default)	r	RES	No Access, All accesses return Bus error
	rh	TRSCTRL	

## Signal Processing Unit 2 (SPU2)

**Table 1012 Reset Values of USROTC**

Reset Type	Reset Value	Note
Power On Reset	0000 0000 <sub>H</sub>	"Power on Reset"

### Access Enable Register 0

The Access Enable Register 0 controls access for writes to registers and configuration memory using the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The SPU is prepared for a 6 bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ... ,EN31 -> TAG ID 011111B.

#### ACCENO

##### Access Enable Register 0

(007E4<sub>H</sub>)

Reset Value: [Table 1014](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rw															

Field	Bits	Type	Description
ENx (x=0-31)	x	rw	<b>Access Enable for Master TAG ID x</b> This bit enables access to the SPU register addresses for transactions with the Master TAG ID x 0 <sub>B</sub> <b>RO</b> , Bus error on write access. 1 <sub>B</sub> <b>RW</b> , Write and read accesses will be executed

**Table 1013 Access Mode Restrictions of ACCENO sorted by descending priority**

Mode Name	Access Mode		Description
Safety ENDINIT	rw	ENx (x=0-31)	Write access permitted during safety initialisation only. Write Accesses at other time errored
Otherwise (default)	r	ENx (x=0-31)	Default Access Mode (Bus Error on Write)

**Table 1014 Reset Values of ACCENO**

Reset Type	Reset Value	Note
Application Reset	FFFF FFFF <sub>H</sub>	Application Reset

### Access Enable Register 1

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). These tags are not used in this system and so no programmable bits are provided.

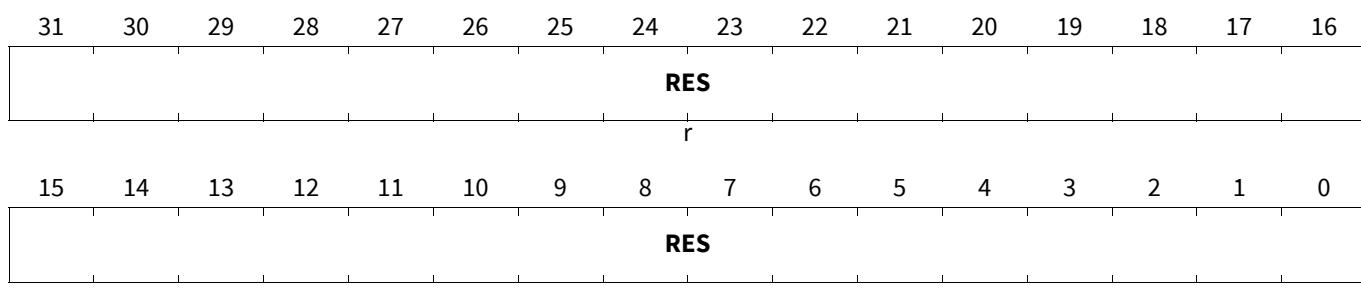
## Signal Processing Unit 2 (SPU2)

### ACCEN1

#### Access Enable Register 1

(007E8<sub>H</sub>)

Reset Value: Table 1016



Field	Bits	Type	Description
RES	31:0	r	<b>Reserved</b> Read as 0, should be written with 0.

Table 1015 Access Mode Restrictions of ACCEN1 sorted by descending priority

Mode Name	Access Mode		Description
Safety ENDINIT	r	RES	Write access permitted during safety initialisation only. Write Accesses at other time errored
Otherwise (default)	r	RES	Default Access Mode (Writes silently ignored)

Table 1016 Reset Values of ACCEN1

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset

### OCDS Control and Status

The OCDS Control and Status (OCS) register controls the module's behaviour in suspend mode (used for debugging).

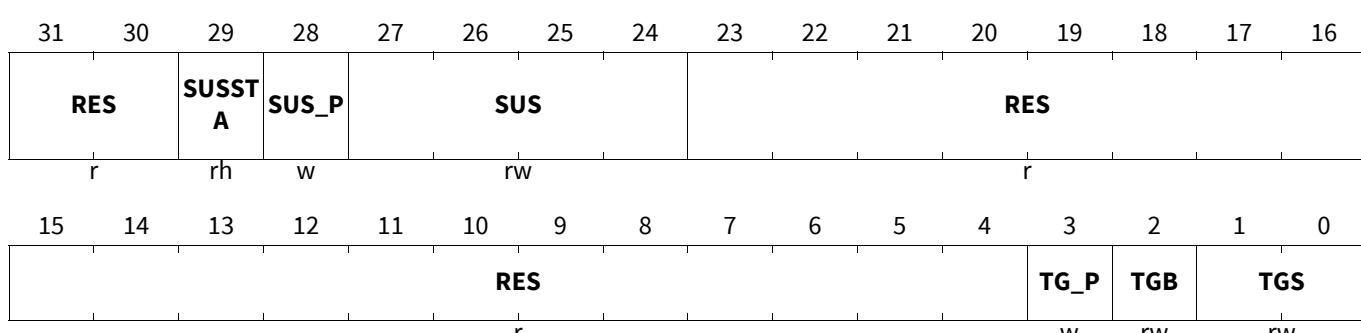
The register can only be written when the OCDS is enabled. While OCDS is disabled, the OCDS suspend control is ineffective .

### OCS

#### OCDS Control and Status

(007EC<sub>H</sub>)

Reset Value: Table 1018



## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
TGS	1:0	rw	<b>Trigger Bus Select</b> Select which of the two possible trigger busses are connected to the output 00 <sub>B</sub> <b>OFF</b> , No Trigger Set Output 01 <sub>B</sub> <b>SETA</b> , Set A Output Triggers are driven onto the output 10 <sub>B</sub> <b>SETB</b> , Set B Output Triggers are driven onto the output 11 <sub>B</sub> <b>RES</b> , Reserved Do Not Use
TGB	2	rw	<b>OTGB0/1 Bus Select</b> 0 <sub>B</sub> <b>OTGB0</b> , Trigger Set is Output on OTGB0 1 <sub>B</sub> <b>OTGB1</b> , OTGB1 Trigger Bus is Output on OTGB1
TG_P	3	w	<b>TGS, TGB Write Protection</b> TGS and TGB are written only when TG_P is 1, otherwise unchanged. TG_P always reads 0
RES	23:4, 31:30	r	<b>Reserved</b>
SUS	27:24	rw	<b>Suspend</b> Enable Suspend of the SPU on Debug System Break. All values for this field not explicitly enumerated will cause a soft suspend of the SPU on break 0 <sub>H</sub> <b>RUN</b> , Break has no effect on SPU operation 1 <sub>H</sub> <b>HARD</b> , Break causes hard suspend of SPU
SUS_P	28	w	<b>Suspend Protect</b> SUS bitfield can only be updated if this bit is set to 1 for the write access
SUSSTA	29	rh	<b>Suspend Status</b> This field will be set to one if the SPU is suspended

Table 1017 Access Mode Restrictions of OCS sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN and OCDS Is Enabled and Supervisor Mode	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rh	SUSSTA	
	w	SUS_P, TG_P	
Master enabled in ACCEN and OCDS Is Enabled and Supervisor Mode and write 1 to SUS_P	rw	SUS	Write Access for permitted masters only, writes for non-permitted masters cause bus error

## Signal Processing Unit 2 (SPU2)

**Table 1017 Access Mode Restrictions of OCS sorted by descending priority (cont'd)**

Mode Name	Access Mode	Description
Master enabled in ACCEN and Supervisor Mode and OCDS Is Enabled and write 1 to <b>TG_P</b>	rw TGB, TGS	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Supervisor Mode	r RES, SUS, TGB, TGS	Read Access Permitted for Supervisor Mode Accesses
	rX SUS_P, TG_P	
	rh SUSSTA	
Otherwise (default)	r RES, SUS, TGB, TGS	Default Access Mode (Bus Error on Write)
	rX SUS_P, TG_P	
	rh SUSSTA	

**Table 1018 Reset Values of OCS**

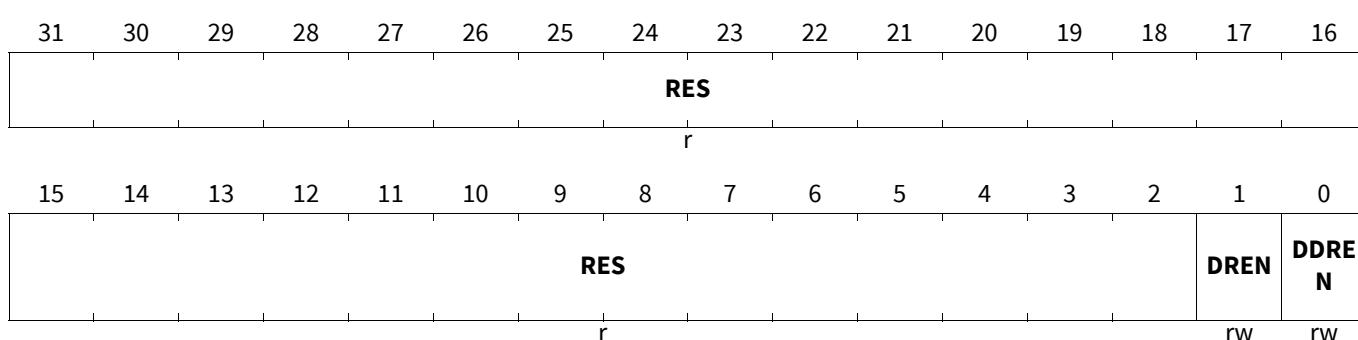
Reset Type	Reset Value	Note
Power On Reset	0000 0000 <sub>H</sub>	"Power on Reset"

### OCDS Debug Access Register

The SPU currently has no registers affected by destructive read. This register is reserved for future extension.

#### ODA

#### OCDS Debug Access Register (007F0<sub>H</sub>) Reset Value: Table 1020



Field	Bits	Type	Description
<b>DDREN</b>	0	rw	<b>Destructive Debug Read Enable</b> If set, reads by the OCDS master to bits which would normally change state on read, do not cause the state to change
<b>DREN</b>	1	rw	<b>Destructive Read Enable</b> If set, reads to bits which would normally change state on read, do not cause the state to change
<b>RES</b>	31:2	r	<b>Reserved</b>

## Signal Processing Unit 2 (SPU2)

**Table 1019 Access Mode Restrictions of ODA sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN and Supervisor Mode	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	DDREN, DREN	
Supervisor Mode	r	DDREN, DREN, RES	Read Access Permitted for Supervisor Mode Accesses
Otherwise (default)	r	DDREN, DREN, RES	Default Access Mode (Bus Error on Write)

**Table 1020 Reset Values of ODA**

Reset Type	Reset Value	Note
Power On Reset	0000 0000 <sub>H</sub>	"Power on Reset"

### Kernel Reset Register 0

The KRST0, KRST1 and KRSTCLR registers are used to initiate and monitor a reset of the SPU kernel logic. The kernel reset is equivalent to a full application reset, only the registers implemented in the bus interface are not affected. All registers affected by kernel reset have an appropriate entry in the "reset values" table

Note, that if a debugger is connected, an application reset of the device will have the same effect on the KRST0 and KRST1 registers as a kernel reset. If KRST0.RST or KRST1.RST are set, they will be cleared and the KRST0.RSTSTAT bit will be set.

This instance of the kernel reset interface registers, KRST0, KRST1 and KRSTCLR, is an exception to the general rule that the kernel reset functionality is protected by ENDINIT. This instance is protected only by the ACCEN protection mechanism.

### KRST0

Kernel Reset Register 0																Reset Value: Table 1022	
(007F4 <sub>H</sub> )																	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
RES																	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	RSTSTAT	RST
																rh	rwh
RES																	

## Signal Processing Unit 2 (SPU2)

Field	Bits	Type	Description
RST	0	rwh	<p><b>Kernel Reset</b></p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to 0B) after the kernel reset is executed</p> <p><math>0_B</math> <b>Run</b>, Normal Operation  <math>1_B</math> <b>Reset</b>, Request a kernel reset of the SPU</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to 0B) after the kernel reset is executed</p>
RSTSTAT	1	rh	<p><b>Kernel Reset Status</b></p> <p>This bit indicates whether a kernel reset was executed or not. This bit is set after the execution of a kernel reset in the same clock cycle both reset bits are cleared (KRST0.RST and KRST1.RST).</p> <p><math>0_B</math> <b>Run</b>, No reset has occurred  <math>1_B</math> <b>Reset</b>, A kernel reset was executed</p>
RES	31:2	r	<p><b>Reserved</b></p> <p>Will always read as <math>0_B</math>. Should be written with <math>0_B</math></p>

**Table 1021 Access Mode Restrictions of KRST0 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rh	RSTSTAT	
	rwh	RST	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	RST, RSTSTAT	

**Table 1022 Reset Values of KRST0**

Reset Type	Reset Value	Note
Application Reset	$0000\ 0000_H$	Application Reset

### Kernel Reset Register 1

The KRST0, KRST1 and KRSTCLR registers are used to initiate and monitor a reset of the SPU kernel logic. The kernel reset is equivalent to a full application reset, only the registers implemented in the bus interface are not affected. All registers affected by kernel reset have an appropriate entry in the "reset values" table

Note, that if a debugger is connected, an application reset of the device will have the same effect on the KRST0 and KRST1 registers as a kernel reset. If KRST0.RST or KRST1.RST are set, they will be cleared and the KRST0.RSTSTAT bit will be set.

This instance of the kernel reset interface registers, KRST0, KRST1 and KRSTCLR, is an exception to the general rule that the kernel reset functionality is protected by ENDINIT. This instance is protected only by the ACCEN protection mechanism.

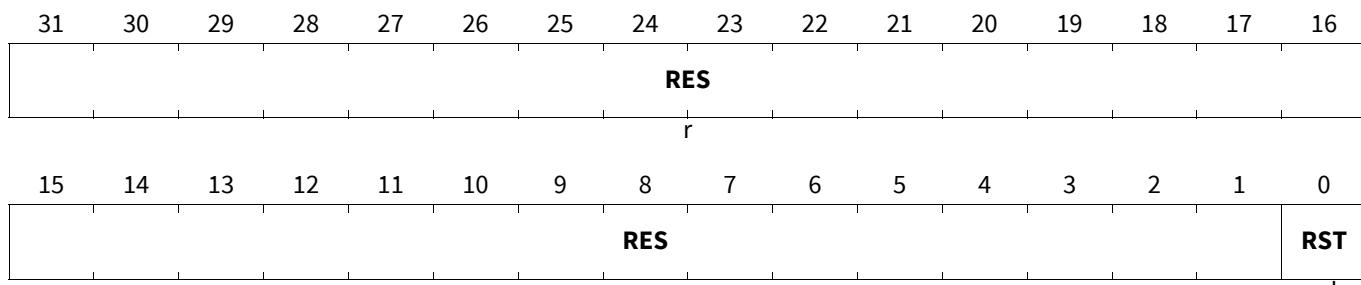
## Signal Processing Unit 2 (SPU2)

### KRST1

#### Kernel Reset Register 1

(007F8<sub>H</sub>)

Reset Value: [Table 1024](#)



Field	Bits	Type	Description
RST	0	rwh	<p><b>Kernel Reset</b></p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to 0B) after the kernel reset is executed</p> <p>0<sub>B</sub> <b>Run</b>, Normal Operation 1<sub>B</sub> <b>Reset</b>, Request a kernel reset of the SPU</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to 0B) after the kernel reset is executed</p>
RES	31:1	r	<p><b>Reserved</b></p> <p>Will always read as 0<sub>B</sub>. Should be written with 0<sub>B</sub></p>

**Table 1023 Access Mode Restrictions of KRST1 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rwh	RST	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rh	RST	

**Table 1024 Reset Values of KRST1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset

### Kernel Reset Clear

The KRST0, KRST1 and KRSTCLR registers are used to initiate and monitor a reset of the SPU kernel logic. The kernel reset is equivalent to a full application reset, only the registers implemented in the bus interface are not affected. All registers affected by kernel reset have an appropriate entry in the "reset values" table

Note, that if a debugger is connected, an application reset of the device will have the same effect on the KRST0 and KRST1 registers as a kernel reset. If KRST0.RST or KRST1.RST are set, they will be cleared and the KRST0.RSTSTAT bit will be set.

## Signal Processing Unit 2 (SPU2)

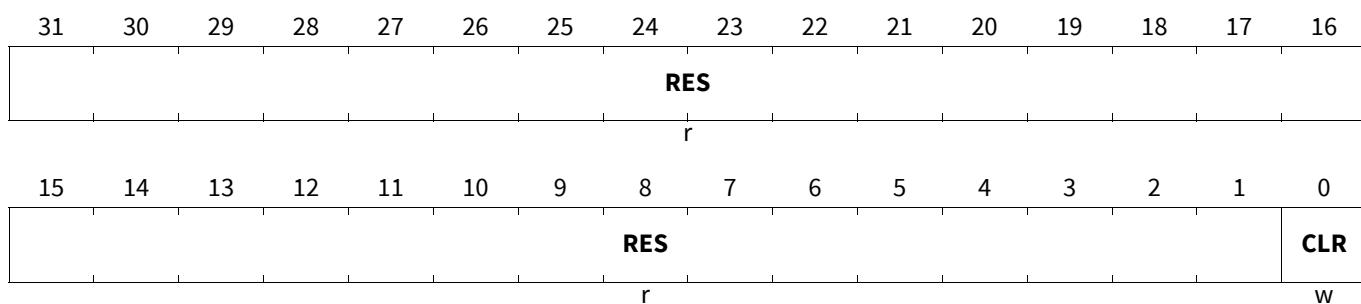
This instance of the kernel reset interface registers, KRST0, KRST1 and KRSTCLR, is an exception to the general rule that the kernel reset functionality is protected by ENDINIT. This instance is protected only by the ACCEN protection mechanism.

### KRSTCLR

#### Kernel Reset Clear

(007FC<sub>H</sub>)

**Reset Value: Table 1026**



Field	Bits	Type	Description
CLR	0	w	<b>Kernel Reset Status Clear</b> Write 1 <sub>B</sub> to clear the KRST0.RSTSTAT bit. Always reads as 0 <sub>B</sub> . 0 <sub>B</sub> <b>Run</b> , No Action 1 <sub>B</sub> <b>Clear</b> , Clear Kernel Reset Status (KRST0.RSTSTAT)
RES	31:1	r	<b>Reserved</b> Will always read as 0 <sub>B</sub> . Should be written with 0 <sub>B</sub>

**Table 1025 Access Mode Restrictions of KRSTCLR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	w	CLR	
Otherwise (default)	r	RES	Default Access Mode (Bus Error on Write)
	rX	CLR	

**Table 1026 Reset Values of KRSTCLR**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset

## Signal Processing Unit 2 (SPU2)

### 20.5 Debug

For debugging purposes, it is possible to stop execution of a linked list. To avoid additional complexity, this can be done by software only before activating the next entry. However the debugger can stop execution using the system wide OCDS suspend functionality.

For availability reasons, the command to stop execution requires to write 3 bits,  $111_B$ , defined as STOP, to the CTRL.MODE bitfield

To reduce complexity, the Radar sequencer can only be stopped by the debugger. It is not possible to resume it. Once the OCDS suspend has been removed, the TC3Ax SPU should be initialised using kernel reset.

Two types of suspend are available

- Hard Suspend: this stops the kernel clock of the TC3Ax SPU immediately. Registers can still be read but should not be written. Config RAM cannot be accessed<sup>1)</sup>. EMEM control signals may suspend in the active state<sup>2)</sup>.
- Soft Suspend: The “GO” signals for the processing modules of the TC3Ax SPU are suppressed. The modules will complete, as far as possible<sup>3)</sup>, processing of the current dataset. Suspend will be acknowledged as soon as the Output Data Manager has completed all outstanding writes to Radar Memory<sup>4)</sup>.

#### 20.5.1 Trace Format

The TC3Ax SPU has a trace connection to the MCDS. The intention of the trace data is to allow synchronisation of TC3Ax SPU output events with CPU activity in the microcontroller. The MCDS has insufficient bandwidth to allow full trace of all FFT results written to EMEM so the data which can be traced simultaneously is limited. The trace logic contains sufficient buffering to allow any four<sup>5)</sup> of the nine available channels to be monitored. The following information can be output to the trace port.

**Table 1027 TC3Ax SPU Trace Data**

Output DMA Channel	Function	Trace Data
1	FFT Results Output	Address and the first 256 bit word written for each FFT processed
2	Signal Power	Address and the first 256 bit word written for each FFT processed
3	Statistical Data	Address and data trace of all values written.
4	CFAR CA Engine Data	Address and data trace of all values written.
5	CFAR Go Engine Data	Address and data trace of all values written.

- 1) The hard suspend function operates by stopping the kernel clock of the TC3Ax SPU. As the config RAM arbitration logic also runs off the kernel clock, the config RAM is not accessible during hard suspend.
- 2) In this case, the EMEM tile control can be used to error the stuck access and permit debugger access to the affected EMEM tile
- 3) In some cases, data will remain in the FFT accelerator pipeline. In these circumstances the UNLOADER module will be unable to complete processing.
- 4) If the histogram function is enabled, accesses to config RAM by the TC3Ax SPU may continue after soft suspend is acknowledged. This is because the FFT pipeline cannot be guaranteed to complete processing during soft suspend and is therefore not used to gate suspend acknowledgement.
- 5) For very short datasets, the bandwidth requirements for DMA channels 3 to 8 increase significantly and the number of simultaneous channels that can be traced drops. Each packet traced requires four clock cycles of the TC3Ax SPU clock. If the number of cycles to output the trace data exceeds the number of clock cycles required for the MATH2 unit to process the dataset, then trace data will be lost

## Signal Processing Unit 2 (SPU2)

**Table 1027 TC3Ax SPU Trace Data (cont'd)**

Output DMA Channel	Function	Trace Data
6	Integration unit 0 (INT0)	Address and the first 256 bit word written for each FFT processed
7	Integration unit 1 (INT1)	Address and the first 256 bit word written for each FFT processed
8	Local Maximum or Logical Function	Address and data trace of all values written.
9	Label List	Address and data trace of all values written.

Trace Data output is controlled by the USROTC.TRSCTRL bitfield. The nine bits in the field correspond to the nine Output DMA channels in ascending order.

### 20.5.2 Debugger events

The following event triggers are available as hardware events for the debugger via the configuration options in the OCS register.

**Table 1028 OTGB Hardware Triggers (Set A)**

0	TC3Ax SPU Processing Starts -Trigger event defined in CTRL register has been detected
1	TC3Ax SPU has completed processing
2	TC3Ax SPU has completed a configuration load
3	Input DMA Done Signal (data block processing completed)
4	Loader Done Signal (data block processing completed)
5	Unloader Done Signal (data block processing completed)
6	ODP Loader Done Signal (data block processing completed)
7	MATH2 Done Signal (data block processing completed)
8	Output DMA Done Signal (last write completed in EMEM, set once per measurement cycle)
9	Output DMA Done (data block processing completed)
10	
11	
12	
13	
14	
15	

**Table 1029 OTGB Hardware Triggers (Set B)**

0	Input DMA Run-time Error - read error signaled by Radar Memory
1	Loader Unit Run-time Error - out of range address generated or error converting half precision floating point input (NaN or Infinity value detected in input data)
2	Config RAM Run-time error - access aborted due to insufficient bandwidth
3	Fixed exponent format compression error - significant bits lost (UNLOADER and ODM) <sup>1)</sup>

## Signal Processing Unit 2 (SPU2)

**Table 1029 OTGB Hardware Triggers (Set B) (cont'd)**

4	Mean Power FIFO Overflow
5	Output DMA Port 1 Error
6	Output DMA Port 2 Error
7	Output DMA Port 3 Error
8	Output DMA Port 4 Error
9	Output DMA Port 5 Error
10	Output DMA Port 6 Error
11	Output DMA Port 7 Error
12	Output DMA Port 8 Error
13	Output DMA Port 9 Error
14	
15	

1) This check occurs even if compression is disabled. In this case the flag will be set if data would have been lost if compression had been enabled

## 20.6 Safety Measures

The TC3Ax SPU has the following safety mechanisms to assist in its use in safety related applications

### 20.6.1 Hardware Safety Mechanisms

The TC3Ax SPU has the following autonomous hardware safety mechanisms which report failures via SMU alarms

#### 20.6.1.1 Register CRC

The register CRC uses a 32 bit ethernet polynomial which assumes that the register contents are converted to an LSB first sequential data stream. The data stream will be generated using the registers from ID\_CONF to CTRL inclusive. Unused register addresses, and the address of the REGCRC register, itself shall be replaced by 0000<sub>H</sub> in the datastream. Additionally, the CTRL.TRIG bit and CTRL.BUSY bit should be assumed to be 0<sub>B</sub> (value when read) and the value of the PACTR.COUNT bitfield should be assumed to be 0<sub>D</sub> for calculating the reference CRC.

The polynomial shall run periodically once enabled and the CRC generated from the register contents compared with the value stored in the REGCRC.CRC bitfield. If the comparison fails, then an SMU alarm will be generated.

The Register CRC is enabled by writing 10<sub>B</sub> to the **SMCTRL**.REGCRCEN bitfield and disabled by writing 01<sub>B</sub>.

Control and comparison logic of the safety mechanism is replicated using negated logic. Any difference in behaviour between the two sets of logic will result in setting of the **SMSTAT**.SMCTRLSTS bit.

The alarm can be tested by writing an invalid CRC to REGCRC.CRC.

If this condition has triggered an alarm, bit 1 of the **SMSTAT**.SMSTAT bitfield will be set to 1<sub>B</sub>.

#### 20.6.1.2 RIF Interface CRC Check

The data path between the RIF and the TC3Ax SPU is protected by a CRC. The CRC is checked once per ramp. The ramp size is determined from the settings of the ID\_CONF register. The check will take place when the data is written into the internal assembly buffer of the Input Data Manager. A CRC failure will result in an SMU alarm. This mechanism is controlled by the **SMCTRL**.RIFCRC bitfield.

## Signal Processing Unit 2 (SPU2)

The CRC check enabled by writing 10<sub>B</sub> to the **SMCTRL**.RIFCRC bitfield and disabled by writing 01<sub>B</sub>. Writing either of the other two possible values will result in the **SMSTAT**.SMCTRLSTS bit being set

If this condition has triggered an alarm, bit 0 of the **SMSTAT**.SMSTAT bitfield will be set to 1<sub>B</sub>.

This alarm condition can be tested by writing 10<sub>B</sub> to the **SMUSER**.RIFCRCTST bitfield.

### 20.6.1.3 RIF Interface CRC Check for Bypass

The data path between the RIF and the TC3Ax SPU Bypass function is protected by its own CRC, separate from the normal RIF Interface CRC. The CRC is checked once per ramp. The ramp size is determined from the settings of the ID\_CONF register. The check will take place when the data is written into the internal assembly buffer of the Bypass function. A CRC failure will result in an SMU alarm. This mechanism is controlled by the **SMCTRL**.R2BPCRC bitfield.

The CRC check enabled by writing 10<sub>B</sub> to the **SMCTRL**.R2BPCRC bitfield and disabled by writing 01<sub>B</sub>. Writing either of the other two possible values will result in the **SMSTAT**.SMCTRLSTS bit being set

If this condition has triggered an alarm, bit 6 of the **SMSTAT**.SMSTAT bitfield will be set to 1<sub>B</sub>.

This alarm condition can be tested by writing 10<sub>B</sub> to the **SMUSER**.R2BPCRCTST bitfield.

### 20.6.1.4 Bypass Data CRC Check

The data stored by the TC3Ax SPU in Radar memory when bypass is enabled is protected by a CRC. One CRC per RIF interface instance selected by BYPASS\_CTRL.SRCRIFm is generated at the end of the measurement cycle using the same ethernet polynomial used elsewhere in the TC3Ax SPU and written at the next consecutive addresses after the last data word written. The CRC values shall be written in order of the RIF instance number. The end of the measurement cycle is determined from the settings of the ID\_CONF register. The CRCs will be checked by the Input Data Manager when the read back during the bypass reload operation. A CRC failure will result in an SMU alarm.

The CRC check enabled by writing 10<sub>B</sub> to the **SMCTRL**.BPCRC bitfield and disabled by writing 01<sub>B</sub>. Writing either of the other two possible values will result in the **SMSTAT**.SMCTRLSTS bit being set.

If this condition has triggered an alarm, bit 2 of the **SMSTAT**.SMSTAT bitfield will be set to 1<sub>B</sub>.

This alarm condition can be tested by writing 10<sub>B</sub> to the **SMUSER**.BPCRCTST bitfield.

This safety mechanism can be tested by writing a short dataset with a known, bad CRC to Radar memory and triggering a bypass reload operation.

### 20.6.1.5 Radar Memory Control Signal Redundancy

All control signals controlling accesses to Radar Memory are duplicated. The duplicate signals are set to the negated value of the primary control signal. In the event of the two signals becoming inconsistent, it can be assumed that an error has occurred. An inconsistent state is detected by comparators and will cause an SMU alarm.

This check is permanently enabled. If this condition has triggered an alarm<sup>1)</sup>, bit 4 of the **SMSTAT**.SMSTAT bitfield will be set to 1<sub>B</sub>.

### 20.6.1.6 Radar Memory Tile Access Error

Radar Memory tiles are enabled for TC3Ax SPU accesses by configuring the tile control mechanism in the EMEM. An TC3Ax SPU access can be directed to a blocked tile either by a configuration error or a fault in the TC3Ax SPU

<sup>1)</sup> The Error flag returned by the EMEM for an access address error is also a redundant signal but, as this is already a monitor rather than a control, an inconsistency in this pair causes the **SMSTAT**.SMCTRLSTS bit to set and does not generate an alarm

## Signal Processing Unit 2 (SPU2)

logic. In the event of the EMEM reporting that an access has failed because it has been mapped to a blocked tile, this mechanism can be used to trigger an SMU alarm.

The alarm is enabled by writing  $10_B$  to the **SMCTRL.RMTAERR** bitfield and disabled by writing  $01_B$ . Writing either of the other two possible values will result in the **SMSTAT.SMCTRLSTS** bit being set.

If this condition has triggered an alarm, bit 5 of the **SMSTAT.SMSTAT** bitfield will be set to  $1_B$ .

This alarm condition can be tested by writing  $10_B$  to the **SMUSER.RMTAERRTST** bitfield.

### 20.6.1.7 Radar Memory Read Data ECC

Read data returned for a Radar Memory access by the TC3Ax SPU is protected by an ECC computed in the EMEM from the address presented to the RAM and the data output from the RAM. This is checked by the TC3Ax SPU to ensure that the data is correct and has been read from the expected RAM location. A failed check will be signalled via an SMU alarm

The alarm is always enabled.

If this condition has triggered an alarm, bit 3 of the **SMSTAT.SMSTAT** bitfield will be set to  $1_B$ .

This alarm condition can be tested by writing  $10_B$  to the **SMUSER.RDECCTST** bitfield.

### 20.6.1.8 RAM ECC

RAM contents are protected by ECC. Corrupted locations will be corrected if a single bit is in error or detected if two bits are in error. Detected failures will be signalled via an SMU alarm

### 20.6.1.9 RAM Address Signature ROM

The RAM address decode is checked by a signature ROM. Failures in the address decode will be signalled via an SMU alarm

### 20.6.1.10 Access Enable

Write accesses to the registers and configuration memory<sup>1)</sup> are controlled by the Access Enable mechanism. This is configured using the ACCEN0/ACCEN1 registers and uses the master tag used to identify the originating master of each write transaction to permit or deny access to modify the contents of the register or memory.

## 20.6.2 Software Based Safety Mechanisms

The TC3Ax SPU supports the following safety mechanisms which require support from software. Failures are to be reported by the software rather than via the SMU

### 20.6.2.1 Software Based Self Test

A gap occurring between application processing operations can be used to run self test patterns through the TC3Ax SPU to detect faults in the TC3Ax SPU hardware and the interfaces between the TC3Ax SPU and the Radar Memory.

### 20.6.2.2 TC3Ax SPU Execution Time Check

A class of faults exist in the TC3Ax SPU hardware which could cause a deadlock of the TC3Ax SPU control hardware. A software check should be provided which does a “sanity check” on the time needed for the TC3Ax SPU to complete execution and flags an error if execution does not complete within the expected time. This check

<sup>1)</sup> The configuration memory is treated as module configuration information rather than system memory and it is therefore appropriate to apply the same protection as for the registers.

## Signal Processing Unit 2 (SPU2)

would also detect faults in the MMIC or MMIC<->uC interface which cause an interruption in the data stream and therefore a stall in the TC3Ax SPU processing. The TC3Ax SPU contains a hardware counter which can be used to support his check.

### 20.6.2.3 Configuration Memory Content Check

The contents of the configuration memory should be checked once written to ensure that all data has been transferred to the memory correctly

### 20.6.3 Hardware Functionality Supporting Software Safety Mechanisms

The following hardware functionality exists within the TC3Ax SPU to increase the effectiveness of the Software Based Safety Mechanisms

#### 20.6.3.1 Monitor CRC Units

The TC3Ax SPU contains CRC units used to provide a method of checking the activity on key internal interfaces. These include:

- The internal RAM interfaces of the FFT accelerator
- Buffer Ram Interfaces
- Input Data Manager EMEM Interfaces
- Output Data Manager EMEM Interface
- Configuration RAM Interfaces

The CRCs shall be separated into two blocks in the registers space.

- CRCs which are independent of the data being processed. These are enabled using **SMCTRL.CTRLCRCEN**
- CRCS which are dependent on the data being processed. These are enabled using **SMCTRL.DATACRCEN**

The first can be used for providing some detection of soft errors occurring during execution as well as during software based self test. The second can only be used while processing known data during software based self test.

The CRC units are mapped into the TC3Ax SPU register space as read only registers.

The CRC Units have an associated control field (**SMUSER.CINIT**). The CRC units are initialised to the reset value by writing  $10_B$  to **SMUSER.CINIT**.

*Note: Enabling in-line CFAR or local maximum based bin rejection will change the address sequence at the ODM output. This affects the CRC accessed via the CTRL2\_CRC register. If the control based CRC is being used then either the CTRL2\_CRC register must be excluded from the CRC checks or dynamic bin rejection based on the data being processed must not be used.*

#### 20.6.3.2 Redundant Control Logic

All control and status state bits (flip-flops) implementing safety mechanism functionality are duplicated. The duplicate flip-flops are used to store the negated value of the bit. This can be observed directly in the SMCTRL and SMUSER registers. In the event of the state of these bits becoming inconsistent, it can be assumed that either a programming or soft error has occurred. An inconsistent state is detected by comparators and can be observed by polling the SMSTAT.SMCTRLSTS bit.

### 20.6.4 SMU events

The following table list the events that can trigger an SMU alarm. See [Section 20.6.1](#) for a description of the hardware safety mechanisms.

## Signal Processing Unit 2 (SPU2)

**Table 1030 SMU Events**

Event	Definition	Source	Comments
uncorrectable ECC error in Radar memory	Read data from a Radar Memory RAM has been corrupted	MTU	One of Multiple alarms will be generated based on MTU configuration
uncorrectable ECC error in a RAM internal to the TC3Ax SPU	Read data from an TC3Ax SPU RAM has been corrupted		
Register CRC check	Computed Register CRC does not match pre-programmed value	TC3Ax SPU	Alarm Mapping is defined in the SMU Specification
Bypass Data CRC check	Computed bypass data CRC does not match the value read from memory		
Radar Memory Control Logic Failure	The control signals for accessing Radar Memory are implemented as redundant logic. This alarm condition indicates that the TC3Ax SPU has detected an inconsistency in the redundant logic		
RIF Data CRC Check	CRC computed from RIF data stream does not match expected value tagged onto the end of the ramp data		
Radar Memory Access Address Error	An access to the Radar Memory has been mapped to a tile that has not been enabled for TC3Ax SPU accesses by writing to the EMEM tile control registers		
Radar Memory Read Data ECC Check	The read data for Radar Memory accesses is returned with an ECC computed from the access address and data. This is checked and an alarm set if the check fails		

Note: *the SMU allows the user to define the action that will be taken as a result of these events (see SMU specification).*

### 20.6.5 Safety assumptions

The safety assumptions for the Radar signal processing unit are described here.

#### 20.6.5.1 Safety assumptions and safety goals

There are 2 safety cases

- Radar with 1 Radar TC3Ax SPU only for low end applications
- Radar with 2 x Radar SPUs for mid range and high performance radar

## Signal Processing Unit 2 (SPU2)

### 20.6.5.1.1 Case1 = Radar with 1 Radar TC3Ax SPU only for low end applications

In the worse case, the Radar/ECU is involved in decision making for emergency brake or active lane assist, where at system level, customers may find it appropriate to use 2 different sensors to confirm the decision.

The other application use case considered would be where the results from the Radar/ECU application are used only to provide advisory information to the driver.

### 20.6.5.1.2 Case2 = Radar with 2 Radar SPUs only for mid to high end Radar

The radar/AURIX™ here is involved at least in decisions for directly implementing emergency braking or active lane steering without driver intervention. Again, the application may be required to use sensor redundancy at the system level.

### 20.6.5.1.3 Case3 = Radar + sensor fusion with 2 Radar SPUs only for mid to high end Radar

From the Radar/AURIX™ perspective, this case is the same as above.

In addition to this, the TriCores in the Radar/AURIX™ will be involved in doing sensor fusion in order to support high integrity decisions based on data coming from the radar/TC3Ax SPU and independent data coming from an external sensor. From the AURIX™ perspective, this use case is similar to the sensor fusion explained separately.

## 20.6.5.2 Radar Application assumptions

The following constraints can be used in an application in order to implement application oriented safety mechanisms.

### 20.6.5.2.1 Case 1: A decision is never taken on single ADC acquisition

This is the basic principle of FFTs for Radar: the FFT is integrating the noise measured by Radar where random variations are filtered out and where only consistent signal variations are kept.

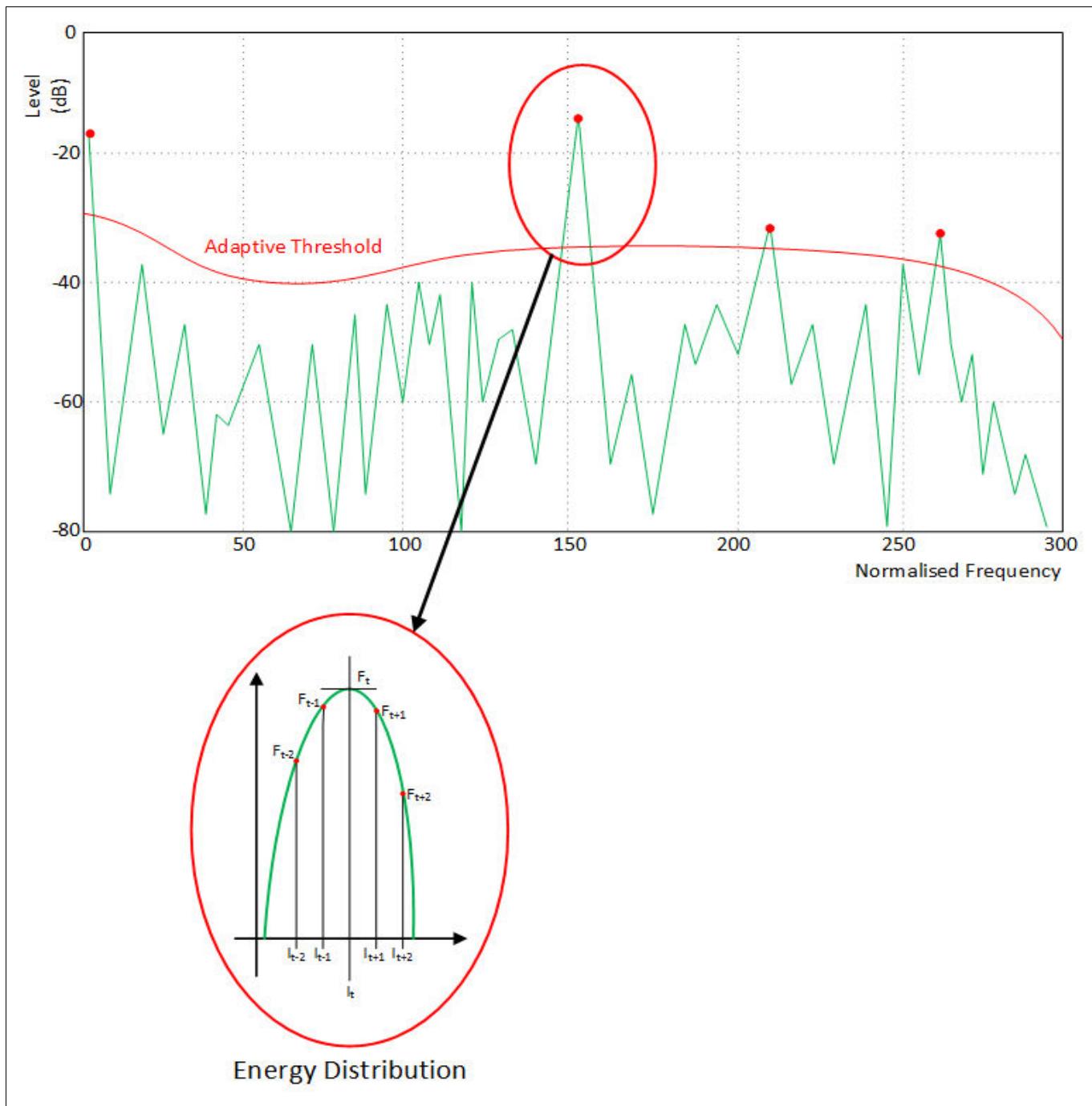
### 20.6.5.2.2 Case 2: A decision is never taken on a single FFT computation (1st stage and 2nd stage FFTs)

- Automotive Radar are now based on multiple antenna organized in phase array. So,
  - Antenna are separated by a distance
  - each antenna has same receiving lobe as the others
  - the difference between antenna is the phase of the signals: this phase difference allows the application to derive the angle of the object from the antenna.

### 20.6.5.2.3 Case 3: FFT peaks are never isolated

The figure below is showing that in fact, energy is distributed to adjacent FFT BINs so that a FFT peak is never isolated

## Signal Processing Unit 2 (SPU2)



**Figure 266 Example of FFT Result Energy Distribution**

A safety mechanism can be implemented to check if an FFT BIN is isolated to reject any isolated FFT BIN (where isolated means that the signal power on adjacent BINs is not high enough and where high enough is defined by application criteria).

Target acquisition is never taken from single acquisition

- this is definitely true for low end (see safety case1)
- this is not true for high performance radar.

Still, even if decision is taken from single acquisition, if we are allowed to assume that the RF device that is doing all preprocessing until the Analog to Digital Conversion is working fine, then, as we have a radar based on multiple

## Signal Processing Unit 2 (SPU2)

antenna with phase difference, an FFT peak leading to target decision is based on valid signal detected by multiple antenna. So

- there should be peaks detected on more than 1 antenna
- each peak on each antenna should meet the energy distribution criteria explained before

### 20.7 Use Cases

See SPU chapter for general assumed use case restrictions. This section covers any specific restrictions for SPU2.

#### 20.7.1 Using Fractional Clock Division (AUXCTRL.DIVF)

This bitfield is used to control the data throughput through the FFT pipeline including LOADER, MATH0, MATH1, FFT and UNLOADER. It is intended to be used to ensure that the FFT data is not passed to the ODP faster than it can be processed by the MATH2 unit. Although the MATH2 unit can exert back pressure on the UNLOADER to prevent overruns, this back pressure is then fed back to the LOADER and IDM. This is not an issue if data is being read from the Radar Memory but could potentially cause unpredictable slipping of input ramps if the data source is the RIF interface.

If data is being read from the RIF and ramp skipping is either intended or expected to be possible, the recommendation is that the AUXCTRL.DIVF bitfield should be set so that the FFT pipeline always takes longer to process the data than the MATH2 unit.

### 20.8 I/O Interfaces

Not in this release.

## Signal Processing Unit 2 (SPU2)

### 20.9 Revision History

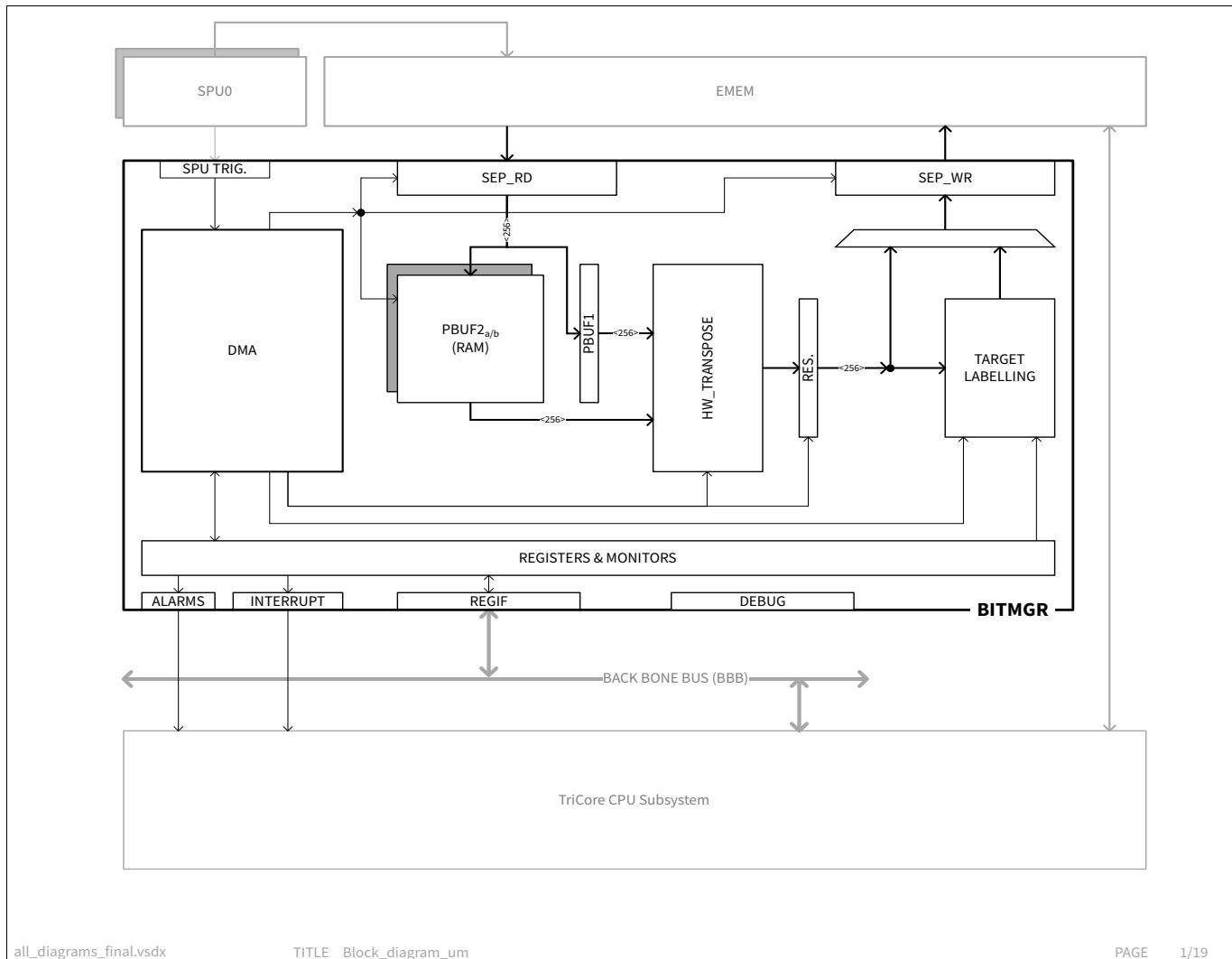
**Table 1031 Document Revision History**

Reference	Change to Previous Version	Comment
<b>V2.0.1</b>		
-	Initial release	
<b>V2.0.2</b>		
-	Rework to clarify new features.	
<a href="#">Page 24</a>	Filter coefficients are Q3.13 format	
<a href="#">Page 26</a>	Auto-DELTA feature (remove DELTA and FACTOR register fields) + modify description of tapering function. Remove LINEAR replacement source and replace with FORWARD (take only output of Forward filter)	
<a href="#">Page 21</a>	Set MATH0 minimum supported sample count to 128 samples	
<a href="#">Page 21</a>	MATH0 DC offset Reciprocal Register	
<a href="#">Page 22</a>	Interference detector replacement	
<a href="#">Page 45</a>	Label address calculation arithmetic increased to 32 bits	

## Bit Manager (BITMGR)

### 21 Bit Manager (BITMGR)

BITMGR provides stand alone semi-autonomous hardware acceleration for bit wise-logical operations on large two dimensional bit-arrays stored in memory. The block is capable of combining two operands or perform reduction operations on a single operand. The operand in context is a two dimensional bit array packed into a wide memory such as EMEM of AURIXPlus.



**Figure 267 BITMGR**

#### 21.1 Feature List

The BITMGR has the following top level features

- A 256 bit RD port to Radar Memory at two cycles per access.
- A 256 bit WR port to Radar Memory at two cycles per access.
- DMA whose capabilities include
  - Read from two dimensional array of user configurable size and aspect ratio.
  - Write to two dimensional array of user configurable size and aspect ratio.
  - Concurrent Read and Write.
  - Two dimensional transfers broken into tiled blocks (non-configurable) size optimized for memory traffic.
  - Write to a circularly indexed region of fixed size

## Bit Manager (BITMGR)

- Simple descriptor chaining for stream processing with minimal supervisory overhead.
- Capability to start on SW or HW triggers and generate HW interrupts and Alarms.
- A logical processing unit capable of
  - transposing one bit array and overlay on another of complementary dimension  
-Or-
  - Combine multiple distinct rows from the same array
- Target labelling; remapping bit array indices to global addresses.

### 21.2 Overview

This section provides an overview of the BITMGR functionality

#### 21.2.1 Glossary of Terms

The following terms are used throughout this document

**Table 1032 Definition of Terms**

Term	Definition
CFAR	Constant False Alarm Rate. A processing operation designed to detect targets in a returned Radar signal compensating for varying amounts of noise in the signal
Transposition	Exchange the co-ordinates of a two dimensional array.
1D	One dimensional, a data structure whose elements can be address with a single index.
2D	Two dimensions, a data structure whose elements can be addressed with two indices.
SPU	Signal Processing Unit
LCLMAX	A short form reference to an algorithm.
DMA	Direct Memory Access

## Bit Manager (BITMGR)



**Start:**

- + Filled dot at the start of a line.
- + Example base address or index(0,0) for a 2D array.



**Contiguous entity:**

- + A contiguous region composed of some basic unit.
- + Example : A string of bits,  
row of a matrix in row major form  
Col of a matrix if in col major form



**Logical partition:**

- + A line comprised of many connected segments, to highlight some logical grouping.
- + Example partitioning of a word, such as adding a zero pad.



**Non contiguous:**

- + A break in a line, or broken line
- + Example: a memory word crossing or bits spread across multiple memory words  
Partition of a matrix.



**Continuation:**

- + Dashed thin line represents continuation.
- + Example access pattern for linear or circular addressing etc.



**Direction:**

- + Example a signal connection or an implication.



**Wide signal:**

- + Example, width of a signal bus.

**Figure 268 Diagram conventions**

### 21.2.2 Functional Overview

Within the context of SPU, the BITMGR post-processes SPU's CFAR and LCLMAX outputs. SPU stores these outputs as two dimensional bit-arrays arranged sequentially in memory. Each bit of the output represents the detection, or otherwise, of a peak in a related FFT bin within a two dimensional range-doppler map.

BITMGR can be configured in the following three modes to accelerate functions on two dimensional bit-arrays. For restrictions on mode usage refer section [Chapter 21.3.4](#)

#### Mode 0, Dual Array Overlay

Logical combination of two, 2D bit arrays of complementary size where the x-axis of one maps to the y-axis of the other and vice-versa.

#### Mode1, Sub-array overlay

Folding of a single, 2D bit array by combining rows at user configurable offsets using logical operations.

#### Mode2, Output Replication

Combine multiple adjacent rows from a single, 2D bit array using logical operations and write the resulting row multiple times to preserve the array size.

#### Target Labelling

In addition to the array processing modes, a post-processing step is also included in BITMGR. Target labelling can be selectively turned on to map valid bit indices in the result to global indices and system addresses, such as the range doppler map. See [Chapter 21.3.3](#)

## Bit Manager (BITMGR)

### 21.3 Functional Description

Though the concept of transpose is simple, its implementation especially on bit matrices remain expensive on most general purpose processors. The time complexity varies according to the size of the memory word, the granularity of the array elements and their packing. BITMGR addresses this problem with a divide and conquer approach to optimize memory bandwidth, throughput and power, details of which are described in this chapter. The performance parameters may be tuned down by altering the clock divider ratios to the IP.

#### 21.3.1 Array operand format

A legal operand to BITMGR shall be a two dimensional data structure in memory conforming to the following description. [Figure 269](#) illustrates the structure view(or matrix view or 2D array) of the required operand in a textbook convention. Rest of this section details how this data structure is stored in memory within the context of BITMGR. The first row shall begin at 256 bit aligned address and forms the base address of the array. Rows (or alternately referred to as the y-axis) are packed in memory. The number of columns per row and the size of a memory word gives rise to two packing modes, categorized as cases 1 and 2. Case 1 applies when the number of columns is greater than the memory word size, and case 2 when the number of columns is less than a memory word size. [Figure 270](#) illustrates case1, where the elements across columns (or alternately referred to as the x-axis) making up each row shall be packed into a multiplicity of 256bit words, contiguous in memory, the last of which may be partial. A partial last word shall be zero-padded to 256bits. [Figure 271](#) illustrates case2 where the elements making up each row shall first be padded to the next power of two size, and such padded rows packed into a 256b word. Therefore in case2, there also exist all zero padded rows following the last valid row. The additional padding thus appends the last row with all-zero rows to form a 256b word. In both cases, the complete two dimensional bit array occupies a contiguous region of memory, with rows packed sequentially in a little endian order within a memory word. The term ‘array-operand’ refers to an instance of this data structure.

BITMGR consumes up to two array-operands simultaneously.

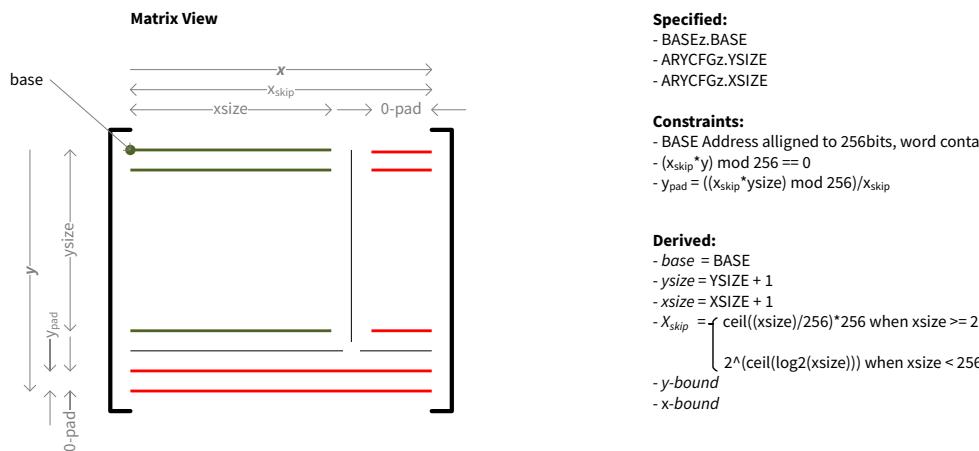
- **Specified:**

- BASE: Base address of Array in EMEM, aligned to 256bit word
- XSIZE: x-size or the columns of the array less 1, excluding zero padding
- YSIZE: y-size or the rows in the array less 1.
- Element granularity: restricted to single bit.

- **Derived:**

- xsize = XSIZE + 1
- ysize = YSIZE + 1
- $X_{skip}$  = The number of bits per row with zero padding.
- x-bound = XSIZE
- y-bound = derived from Mode0, 1 or 2

## Bit Manager (BITMGR)

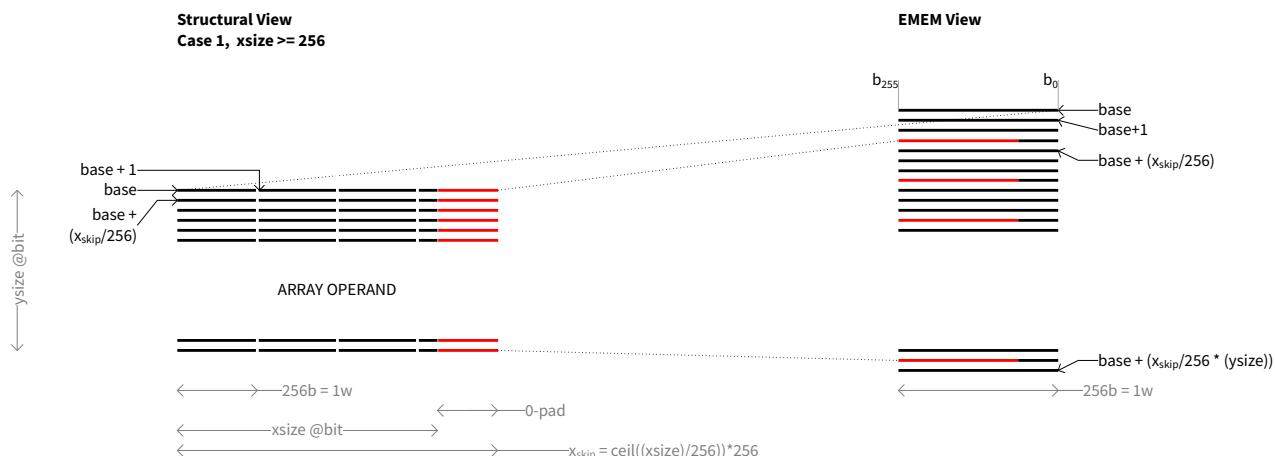


all\_diagrams\_final.vsdx

TITLE Operand\_view

PAGE 1/19

**Figure 269 Array operand and attributes (Matrix/structural/2D array view)**



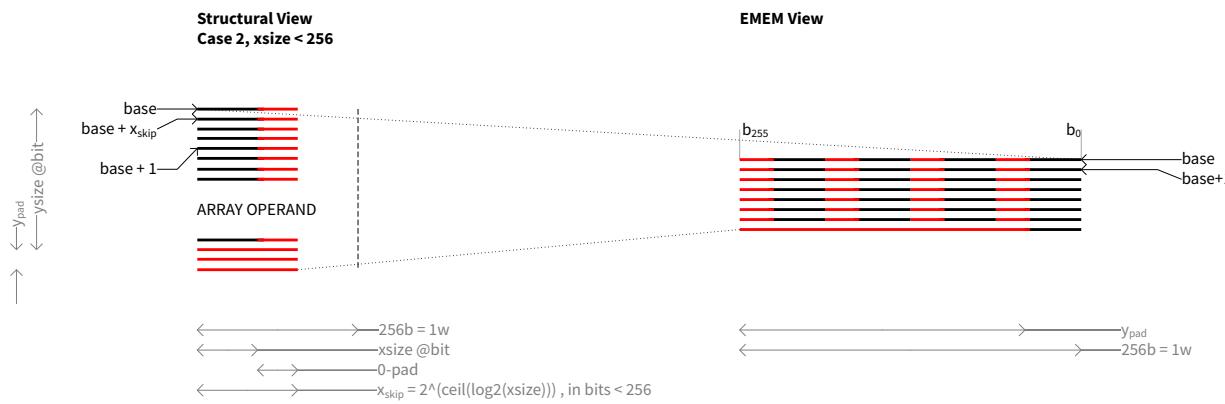
all\_diagrams\_final.vsdx

TITLE Array\_operand\_case1

PAGE 1/19

**Figure 270 Array Operand packing, Case1**

## Bit Manager (BITMGR)



all\_diagrams\_final.vsdx

TITLE Array\_operand\_case2

PAGE 1/19

**Figure 271 Array Operand Packing, Case2**

### 21.3.2 Operations

BITMGR defines three operations on array-operands, categorized as execution Modes 0, 1 and 2. Mode0 and Mode1/2 are executed sequentially in subsequent passes when both are enabled. Only one of Mode1 or 2 can be chosen after Mode0. If chaining is enabled both modes should have the same array operand size. See chaining operations [Chapter 21.3.4](#)

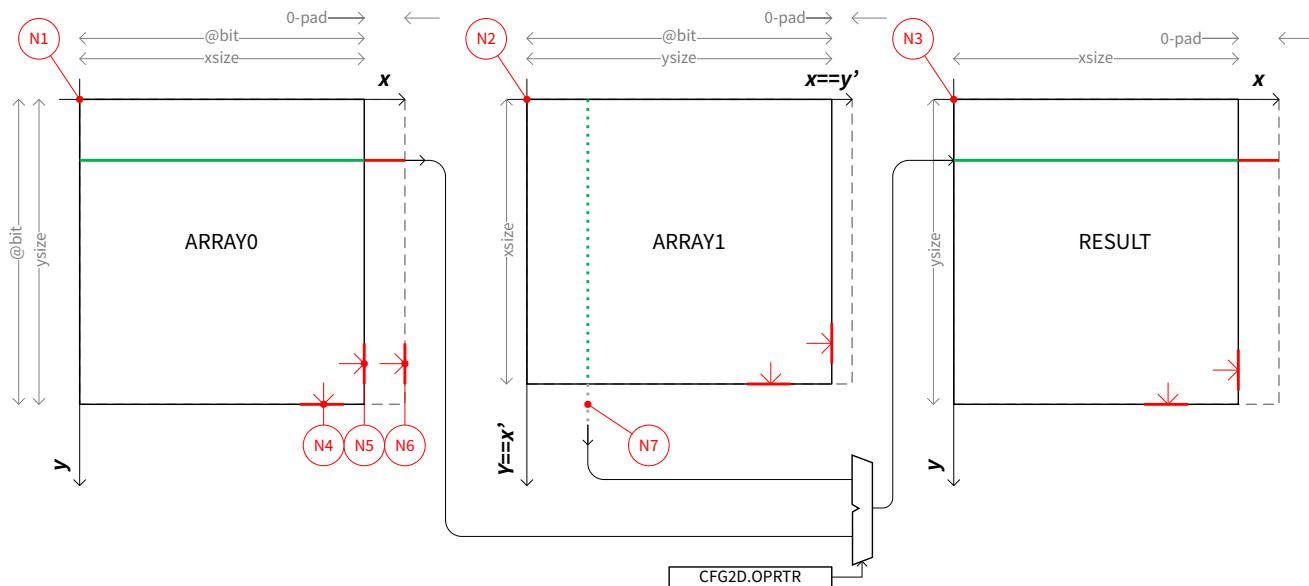
Logical operation is defined on elements of an array-operand where each element is a single bit. Logical operation is configured as a look-up table with the **CFG1D**, **CFG2D** ‘OPRTR’ fields. Lookup table output is indexed LSB to MSB. The binary value of this index shall be 2bits wide represented as {idx<sub>1</sub>, idx<sub>0</sub>}. A single bit from first array-operand shall be used as idx<sub>0</sub> and a corresponding bit from second array-operand shall be used as idx<sub>1</sub>.

#### 21.3.2.1 Mode 0, Dual Array Overlay

Logical combination of two array-operands of complementary size where the x-axis of one maps to the y-axis of the other and vice-versa. See [Figure 272](#)

- Pre-requisites and operation:
  - Enable **CTRL.DOEN**
  - Configure base address of the first array-operand; Array 0(BASE0.BASE)
  - Configure the base address of the second array-operand; Array1(BASE1.BASE)
  - Configure size of Array 0 (ARYCFG0.XSIZE, ARYCFG0.YSIZE). Array1 shall infer its size from ARYCFG0 such that it is complementary to Array0.
  - Configure bitwise logic operation (**CFG2D.OPRTR**) used to combine rows from first operand to corresponding columns on second operand, element wise. All elements from the array-operands shall be combined using the same operation.
  - Configure triggering mechanism using CTRL. Once triggered the process shall run autonomously until the y-bound row on Array 0. See also [Chapter 21.3.5](#)
  - Configure a base address (**DAOBASE.BASE**) to store the RESULT to. The RESULT dimensions shall be inherited from first array-operand.
  - The RESULT conforms to the definition of array-operand.

## Bit Manager (BITMGR)



all\_diagrams\_final.vsdx

TITLE Operational\_mode0

PAGE 1/19

**Figure 272 Mode0, Dual Array Overlay**

Note:

- (1) Base Address of Array0 corresponds to address of first row, of which the first column is implied.
- (2) Base Address of Array1, to be transposed, inherits complementary dimensions from Array0. Note how x and y coordinates are interchanged
- (3) Base address of result. Result inherits dimensions from Array0.
- (4) Array y-bound, beyond which memory is not reserved if the memory word contains only zeros all arising from padding.
- (5) Array x-bound where the 256 bit word can be partial, and is padded with zeroes.
- (6) a legal case1 or case2 packing boundary.
- (7) All zero padded rows may not exist in memory, instead zero extended by the fetch path.

### 21.3.2.2 Mode1, Sub-array overlay

Folding of a single array-operand by combining with a bit wise logical operation, multiple sub arrays of equal dimension placed at different offsets relative to base. The result sub array size is equivalent to the input sub-array.

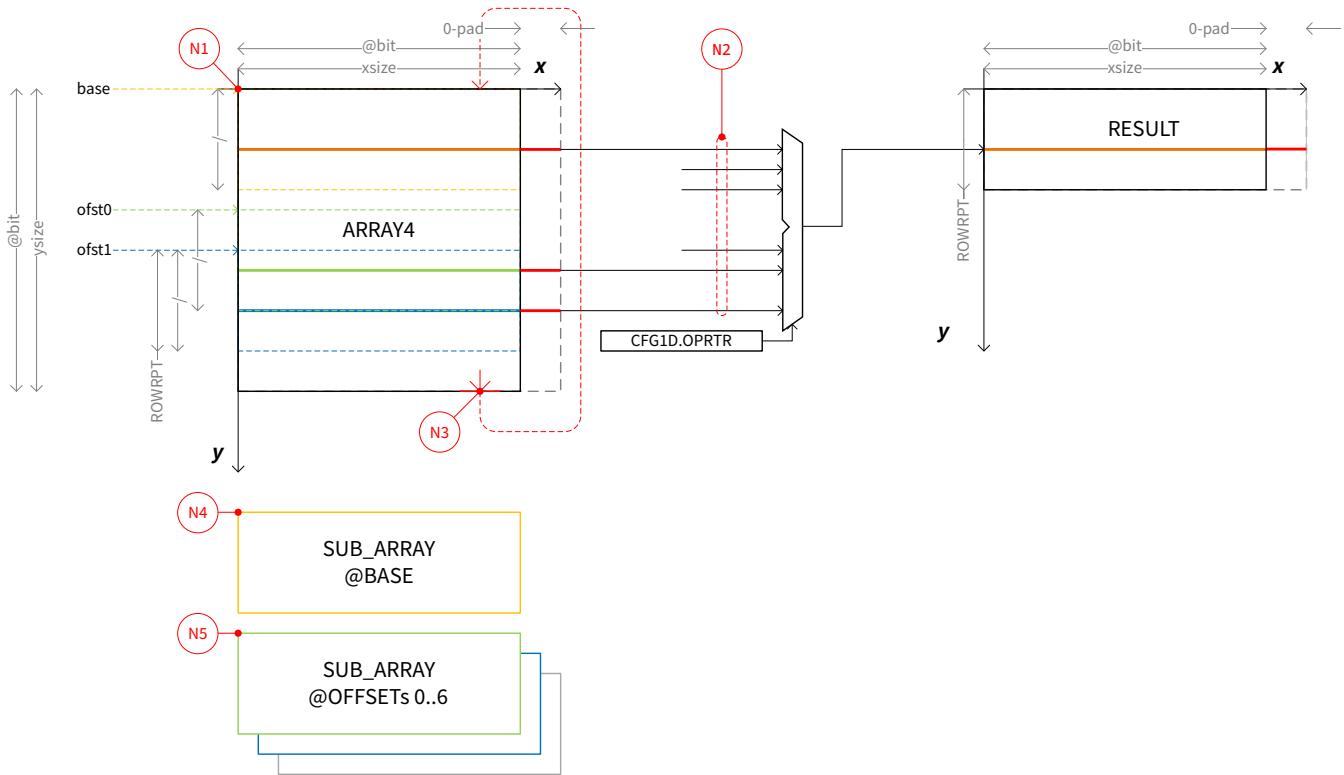
- Pre-requisites and operation:
  - Enable **CTRL.SOEN**
  - Mode 1 is selected over Mode2, when the (**CFG1D.SAF**) bit is  $0_B$ .
  - Configure base address for array-operand Array4 as **BASE4.BASE**.
  - Configure the x-dimension and y-dimension of the array-operand as **ARYCFG1.XSIZE**, **ARYCFG1.YSIZE**. If chaining is enabled, observe chaining mode constraints as detailed in **Chapter 21.3.4**, specifically on **XSIZE**
  - Configure the number of consecutive rows making up the sub arrays as row repeats (**CFG1D.ROWRPT**). This with inherited x-dimension, **XSIZE**, define the shape of all input sub arrays as well as the output.
  - Configure the row offsets to extract sub arrays from (**OFSTv(v=0-6).ROW\_OFST**) and enable the setting in **OFSTv.EN**. All row offsets are implied to align with column 0 of Array4. All sub-arrays have same y-dimension, and are exactly **XSIZE** wide.
  - Configure **CFG1D.IS** bit to include the implicit sub-array defined at **BASE4.BASE** in the computation. The implicit sub-array is included by default. If disabled, **OFST0** shall be configured and enabled. (Not enabling

## Bit Manager (BITMGR)

at least one OFSTv when CFG1D.IS is true, copies the input to output. Not enabling another OFSTv other than OFST0, will copy input to output with a row permutation, i.e a cyclic shift of the input along y.

- The number of sub arrays (2D) to be combined to produce the output (up to a maximum of 1 base+7 offsets, one always from base when included and up to 7 from offsets) shall be inferred from enabled offsets. Co-located row from each sub array is collected into a **row-group**. A **row-group** thus consists of a row from Base sub-array when included and a row each from the enabled offset sub-arrays in increasing order of precedence with respect to sub-array number.
- Configure the bitwise logical operation (**CFG1D.OPRTR**) used to combine the data from the **row-group**. Elements in the sub-arrays shall be combined using the same logical operation.
- On starting a fresh **row-group**, the second operand shall be from the sub-array pointed to by BASE4(or OFST0), and the first operand shall be from the sub-array pointed to by OFST0(or OFST1) forming the items 1 and 2 respectively in the **row-group**. On every successive row operation in the current **row-group**, the second operand shall be the result of the previous operation and the first operand shall be from remaining items of the **row-group** iterated in order. Here an operand refers to the chosen row.
- Offset sub arrays wrap around at YSIZE. Row access past the last row of the base array, continue from row0.
- ROWRPT should be less than or equal to YSIZE of the corresponding base array. ( Prevents the target labelling function from producing erroneous results)
- If the number of enabled offsets is less than 7, they shall use contiguous **OFSTv (v=0-6)**.
- For the purpose of target labelling, the index chosen corresponds to the index of the implicit sub-array portion or sub\_array from OFST0 depending on CFG1D.IS.
- The RESULT conforms to the definition of Array operand.

## Bit Manager (BITMGR)



all\_diagrams\_final.vsdx

TITLE Operational\_mode1

PAGE 1/19

**Figure 273 Mode 1, Sub-Array overlay**

Note:

(1) Base address, for array to be processed in mode1 (2) Combine multiple rows from the same array into a single output using the configured logical operation. illustrates row-group ordered top to bottom in figure (three rows shown, maximum seven configurable) (3) If the access crosses the y-bound, rows shall be fetched starting at row 0, i.e. cyclically (4) The sub array derived from base is selectively included in the operation. (5) up to 7 additional sub-arrays derived from the various configured offset can be included in the operation. All sub arrays (including base) have the same dimension. Figure depicts a y-dimension i.e. ROWRPT smaller than the ysize for clarity.

### 21.3.2.3 Mode2, Output Replication

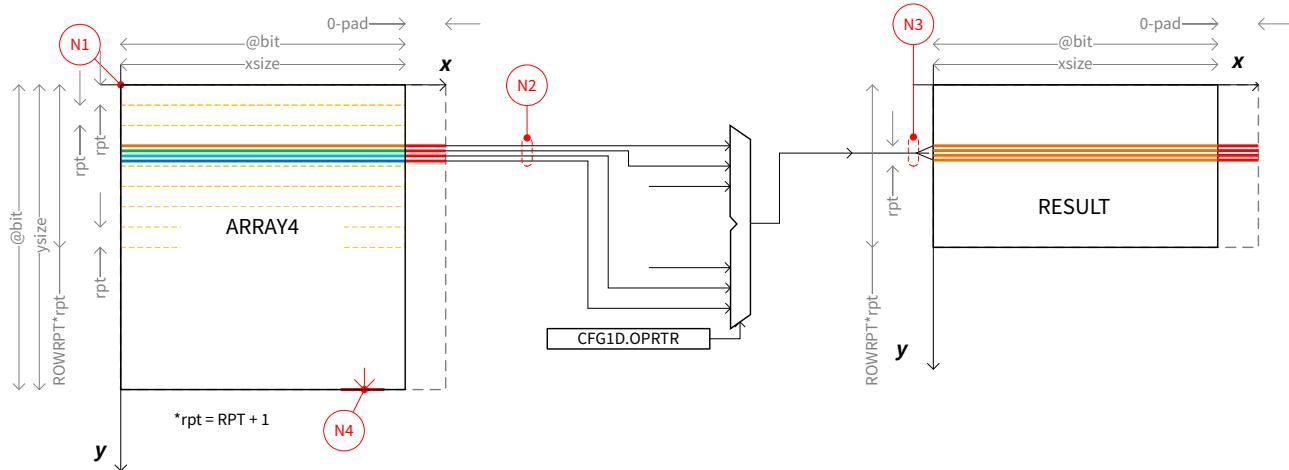
Combine using logical bit wise operations, ‘N’ adjacent rows from a single input array-operand and write the resulting row ‘N’ times. Repeat the above operation ‘M’ times to produce the result.

- Prerequisites and operation:
  - Enable **CTRL.SOEN**
  - Mode 2 is selected over Mode1, when the (**CFG1D.SAF**) bit is  $1_B$ . A RPT number of adjacent rows in ascending order is the **row-group** for Mode2
  - Mode 2 will use the same Array base and dimension as Mode 1.
  - Mode 2 will use the same x-dimension as base Array dimension.
  - Configure number of passes as row\_repeat (**CFG1D.ROWRPT**). ROWRPT provides the number of sub-arrays, such that they abut each other along the y dimension and are non-overlapping. The y-bound is derived from YSIZE. Crossing the y-bound continues row access from row index 0. The ysize of the RESULT

## Bit Manager (BITMGR)

is a function of **CFG1D.RPT**, **CFG1D.ROWRPT** and **ARYCFG1.YSIZE**. Note that only YSIZE rows can be accommodated in the output and in the event the total number of rows to be processed exceeds the YSIZE limit, the operation terminates when YSIZE rows are complete. Thus, the last row-group may be partial on the input, and only a similar number of rows are flushed to memory to form the result, or in other words a row from the input operand shall be used at the most once and the RESULT shall be at the max equivalent in dimension to the input Array-operand.

- The first **row-group** is the group starting at row 0 positioned at BASE if CFG1D.IS is enabled, or the first **row-group** is the group starting at OFST0. It is illegal not to configure and enable OFST0 if IS is 0<sub>B</sub>.
- Configure logical operation used to combine all the data in the **row-group**(**CFG1D.OPRTR**). All elements from the sub-arrays shall be combined using the same logical operation.
- All rows from the **row-group**, are combined to generate a single row result. The sequencing of the individual rows shall follow the combination order as defined in Mode1, where adjacent rows replace offset rows.
- The result generated by the previous step is replicated on the output by a similar number as the rows used in the combination. This implies that if the last row-group is partial, the replication will be less than RPT.
- If access crosses the y-bound they continue from row0, i.e. cyclical.
- Target labelling would use the absolute y-index as used in the row-group.
- The RESULT conforms to the definition of the Array operand.



all\_diagrams\_final.vsdx

TITLE Operational\_mode2

PAGE 1/19

**Figure 274 Mode 2, Output Replication**

Note:

- (1) Mode 2 shares Array parameters with Mode 1. Only one mode can be chosen
- (2) Adjacent rows are combined using configured operator. Illustrates row-group ordered top to bottom
- (3) Result from combining the entire row-group is written out as many times as the size of the row-group in rows.
- (4) Row access that cross y-bound wraparound, but operation terminates if BASE or OFST0 is reached depending on IS.

### 21.3.3 Target Label List Generation

The result from the operand combination steps above can be post processed to map each valid bit index (a non zero element) to a global index and system address. When **CTRL.TLBLE** is configured to a legal value, BITMGR shall enable this inline post process step. The generated list shall be referenced to the co-ordinate axes of the input array-operand (inherited from first array-operand specified at BASE0/2 for Mode0 or first enabled sub-array of array-operand used in Mode1,2 depending on CFG1D.IS).

## Bit Manager (BITMGR)

**Note:** Enabling the Target Label List function will significantly impact performance of the BITMGR since target identification is carried inline with every row of result computed.

For each word in the result, any bit set to one has its co-ordinates in input array computed. These are then written to memory along with an address computed using the **TLBLB.BASE**, **TLBLO.XMUL** and **TLBLO.YMUL** bitfields. If the co-ordinates are **X\_index** and **Y\_index** the address is

```
32b:SysAddress := TLBLB.BASE + (TLBLO.XMUL * X_index) + (TLBLO.YMUL * Y_index)
```

The format of the list generated is derived from the format used by the equivalent function in the SPU. The output consists of two, thirty-two bit words per target. The first word contains the generated system address. The second word contains the X-index in bits 9:0; the Y-index in bits 19:10. As the inputs to the BITMGR are always two-dimensional arrays, Z-axis co-ordinates do not exist. Therefore, Bits 27:20 shall be reused for the running batch count (**CTRL.PCNT\_STAT**) when batch mode processing is in progress. Bit 31 is used to flag a valid label. The list would mostly contain valid labels. However, at the end of every iteration in the batch or a single, a partial list entry could exist. In this instance, a list entry with at least one label is flushed to memory. The valid bit avoids ambiguity and optionally provides a quick method to test the validity of a target rather than testing address bounds.

```
32b:Label_ew := SysAddress
32b:Label_ow := {1, 3'b000, CTRL.PCNT_STAT, Y_index, X_index}
64b:Label_in := {Label_ow, Label_ew} // nth target
256b:Label_list_item := {Label_i3, Label_i2, Label_i1, Label_i0}
```

When a target list entry(i.e containing up to 4 targets) is written to radar memory, the value in the **TCNT.COUNT** bitfield is incremented by 1. **TCNT.COUNT** is initialised to  $0_D$  when the **CTRL** register is written. **TCNT.COUNT** is 16 bits, where bits 0 to 9 point to the current list index, and the higher order bits provide the number of roll-overs since initialization.

### 1D Circular addressing

The maximum size of the Target List shall be 32 KiB; enough to store 4096 targets. The BITMGR will treat the target list as a circular buffer and will reset the output address to the address stored in **TLBLBASE.BASE** every time the value of **TCNT.COUNT** reaches an integer multiple of 4096. To allow application software to manage the buffer, interrupts shall be generated at 50 and 100% buffer watermarks; i.e. **TCNT.COUNT** reaches multiples of 2048. Target generation, when enabled, will add an extra DMA inline with every 1x256 bit of result computed. The DMA uses a dedicated descriptor with 1D Circular mode. Writes are always 1x256bits. The only exception is at the end of configured processing or completion of every iteration within a batch, where a partial 256bit word might need to be flushed to complete the operation.

#### 21.3.4 Function triggers and function chaining

This sections details the control options within BITMGR used for selecting triggers, selecting functions, their sequencing and interdependencies between configured operations. The term ‘unit of work’ is broadly used to collect autonomous actions that follow a trigger event. It is illegal to change configuration parameters in an armed or triggered state.

##### 21.3.4.1 Independent operation of functions

**CTRL.DOEN** and **CTRL.SOEN** enable Dual Array overlay processing (Mode0) and Sub-Array overlay processing (Mode1/2) respectively. They can each be enabled independently or both together. If Mode0 is disabled, a trigger will initiate Mode1/2 without delay. However only one of Mode 1 or 2 is selected at a time by **CFG1D.SAF**. Also refer

## Bit Manager (BITMGR)

Array-operand pointer management in the section on **Batch mode trigger**. If only one function is enabled, the value of the CTRL.CHAIN bit should be ignored.

**Note:** *The chaining operation presents numerous structural hazards, and hence shall be blocking in that Mode0 and Mode1/2 will not be pipelined. Chaining impose further restrictions on operand dimension when **Run mode** is set to “CONTINUOUS”. Target labelling can be enabled either on Mode0 result or Mode 1/2 result. However in chained mode, Mode0 result area will be reused on every trigger.*

### 21.3.4.2 Function chaining

Chaining is active when **CTRL.CHAIN** and both **CTRL.DOEN** and **CTRL.SOEN** are enabled. This facilitates the output from a Dual Array Overlay Processing(Mode0) to be used as the input to a Sub-Array Overlay Processing (Mode1/2). However there shall be no internal forwarding of results between the two function. Instead, a fully formed operand array in EMEM shall be passed into the second function by configuring BASE4 and 5, ARYCFG1 to be the same as **DAOBASE** and ARYCFG0 respectively. DAOBASE shall not increment when chaining is enabled. Also refer Array-operand pointer management in the section on **Batch mode trigger**

Only one sub-array processing mode (mode1/2) can be chosen with **CFG1D.SAF**.

### 21.3.4.3 SW Trigger

Setting **CTRL.T\_SRC=SWTRIG** triggers BITMGR on completion of a TRIG register write with TRIG.EVENT set. If a MODE1/2 and MODE0 are both enabled, the sequence is executed by a trigger event. Trigger shall start Mode0, completely form its output in memory, and internally trigger Mode1/2 without further software intervention. Receiving a second trigger from software (when **CTRL.T\_SRC** sets software trigger) before the completion of an ongoing operation shall trigger an ERROR interrupt. The assumption is that untimely SPU activity corrupts input array-operands worked on by the BITMGR. BITMGR shall halt processing on this event, and ensure that its interfaces are transitioned to a benign state so as not to interfere with the rest of the system. Even on ERROR, TRIG.BUSY continues to reflect internal state of the module, such as pending Radar Memory writes for example.

### 21.3.4.4 HW Trigger

Alternately, BITMGR can be triggered with a SPU DONE signal by setting **CTRL.T\_SRC** to SPU0 or SPU1. On completion of TRIG register write with TRIG.EVENT set, BITMGR shall be armed, waiting for a corresponding DONE signal from the configured SPU to trigger processing. Both sequencing and monitoring operations work similar to that of the SW Trigger scheme.

## 21.3.5 Run mode

Triggers can be configured for single or batch mode processing of operands

### 21.3.5.1 One-Shot trigger

**CTRL.T\_TYPE** shall be set to single and **CTRL.T\_SRC** not OFF. The default triggering mode is fundamental to interrupt driven processing where BITMGR shall perform a single (Independent set or chained set) operation and halt. The BITMGR can be configured to generate an “ATTENTION” interrupt when processing completes. Operands configured at BASE0, BASE1 shall be used for Mode0 and BASE4 for Mode1/2.

If a Mode1/2 is chained with a Mode0 the chained sequence is executed by a trigger event. Trigger shall start Mode0, completely form its output in memory, and internally trigger Mode1/2 without further software intervention. The result is the output of Mode1/2 operation.

Result shall be placed at **DAOBASE** or **SAOBASE** or both. A previous result may be overwritten following a trigger.

## Bit Manager (BITMGR)

**Note:** One-shot trigger may be interpreted as a subset of batch mode trigger where the batch size is preset to one.

### 21.3.5.2 Batch mode trigger

Batch mode triggering, **CTRL.T\_TYPE** = “CONTINUOUS”, configures BITMGR to autonomously process a stream of array-operands. While it does work when **CTRL.T\_TYPE**=SWTRIG, the intent is to use it with **CTRL.T\_SRC**=SPU0 or **CTRL.T\_SRC**=SPU1 where BITMGR shall be triggered directly by the appropriate SPU DONE signal. A trigger is still required to advance the stream until a batch size of **CTRL.PCNT**.

- If the Target Labelling Function is enabled, then bits 27:20 of the second word shall contain a running count of the processing operation performed. (See the definition of the **CTRL.PCNT\_STAT** bitfield)
- The BITMGR shall only generate an ATTENTION interrupt at the completion of a number of processing operations as configured by **CTRL.PCNT**.
- Receiving a DONE input from the SPU (when **CTRL.T\_SRC** sets hardware trigger) or a new SW Trigger before the completion of an ongoing operation shall trigger an ERROR interrupt. The assumption is that untimely SPU activity corrupts input array-operands worked on by the BITMGR. BITMGR shall halt processing on such an event and ensure that its interfaces are transitioned to a benign state so as not to interfere with the rest of the system. Even on ERROR, TRIG.BUSY continues to reflect internal state of the module, such as pending Radar Memory writes for example.

Batch mode triggering shall manage input and result pointers as follows.

#### Mode 0 Mode1/2 independent operation

If the Dual Array Overlay Processing Mode (Mode0) is enabled, the base addresses for the two input array operands shall alternate between BASE0 (for Array 0), BASE1 (for Array 1) and BASE2, BASE3 respectively on each successive HW trigger. The configuration facilitates a ping pong buffer in EMEM, pipelining array operands between SPU and BITMGR (not to be confused with the private buffer inside BITMGR). The first processing operation after configuration shall be BASE0.BASE and BASE1.BASE.

The EMEM address used for writing results shall continuously increment starting from **DAOBASE**.BASE. There shall be **CTRL.PCNT** number of output operand-arrays in contiguous memory with the first positioned at **DAOBASE** (an array of array-operands). [Figure 269](#) depicts the valid packing for elements (each an array-operand) within an array of array-operands. Every result must conform to the Array-operand definition and hence should be placed at a base address which is aligned to a 256bit EMEM address.

If the Single Array Overlay Processing Mode (Mode1/2) is enabled, the base addresses for the two input array operands shall alternate between BASE4 and BASE5, respectively on each successive HW trigger. The configuration facilitates a ping pong buffer in EMEM, pipelining array operands between SPU and BITMGR (not to be confused with the private buffer inside BITMGR). The first processing operation after configuration shall be BASE4.BASE.

The EMEM address used for writing results shall continuously increment starting from **SAOBASE**.BASE. There shall be **CTRL.PCNT** number of output array-operands in contiguous memory with the first positioned at **SAOBASE** (an array of array-operands). [Figure 269](#) depicts the valid packing for elements (each an array-operand) within an array of array-operands. Every result must conform to the Array-operand definition and hence should be placed at a base address which is aligned to a 256bit memory address.

#### Mode0 chained with Mode1/2

Batch processing of operand stream in Mode0 continues as above. However, an important distinction is that the position of result array-operand remains fixed to **DAOBASE**.BASE i.e. the results shall be overwritten on every subsequent trigger.

## Bit Manager (BITMGR)

BASE4/5 shall point to **DAOBASE**, so as to pass the result from Mode 0 to Mode1/2 through memory. It is also required that ARRAYCFG0 and ARRAYCFG1 are configured to the same values.

The result array from Mode1/2 will now be placed at incrementing addresses starting from **SAOBASE.BASE**. The next result array will be placed starting at the next contiguous location. There will be exactly **CTRL.PCNT** number of result arrays when the batch process completes. Every result must conform to the Array-operand definition and hence should be placed at a base address which is aligned to a 256bit boundary.

### 21.3.6 Interrupts

The BITMGR generates two interrupts:

- **ATTENTION** will be generated under the following conditions
  - If **CTRL.T\_TYPE=ONESHOT**, the interrupt will be generated when the configured processing operations have completed following one trigger.
  - If **CTRL.T\_TYPE=CONTINUOUS**, the interrupt will be generated if **CTRL.PCNT** is not equal to  $0_D$  and the enabled processing operations have completed following **CTRL.PCNT** triggers.
  - To allow application software to manage the target buffer, the BITMGR will generate interrupts when the value of **TCNT.COUNT** reaches integer multiples of 2048 i.e. every time the buffer reaches 50% or 100% of capacity
- **ERROR** is used to signal that an event has occurred that either potentially or actually has corrupted the execution results.
  - If a trigger is received before the completion of an ongoing operation triggered by the last trigger, an ERROR may be flagged. The trigger event monitored is a HW or SW trigger event depending on the **CTRL.T\_SRC** setting. Note that target labelling in continuous mode could create error conditions as the compute time is no longer deterministic.
  - When the ERROR interrupt is raised, the default behaviour of BITMGR is to halt and transition all its interfaces to a benign idle state so as not to interfere with the rest of the system. **TRIG.BUSY** continues to reflect internal state of the module, such as pending Radar Memory writes for example.

**Table 1033 Bit map for Interrupt mask register STATCTRL.INTMASK**

Event	Description	Note
End of execution	End of operation, <b>CTRL.T_TYPE==”ONESHOT”</b>	STATCTRL.INTMASK[0]
End of execution	End of PCNT operations, <b>CTRL.T_TYPE==”CONTINUOUS”</b>	STATCTRL.INTMASK[1]
Target list @2048 entries	Target list crossed half watermark	STATCTRL.INTMASK[2]
Target list @4096 entries	Target list crossed full watermark	STATCTRL.INTMASK[3]

**Table 1034 Bit map for Error mask register STATCTRL.ERRMASK**

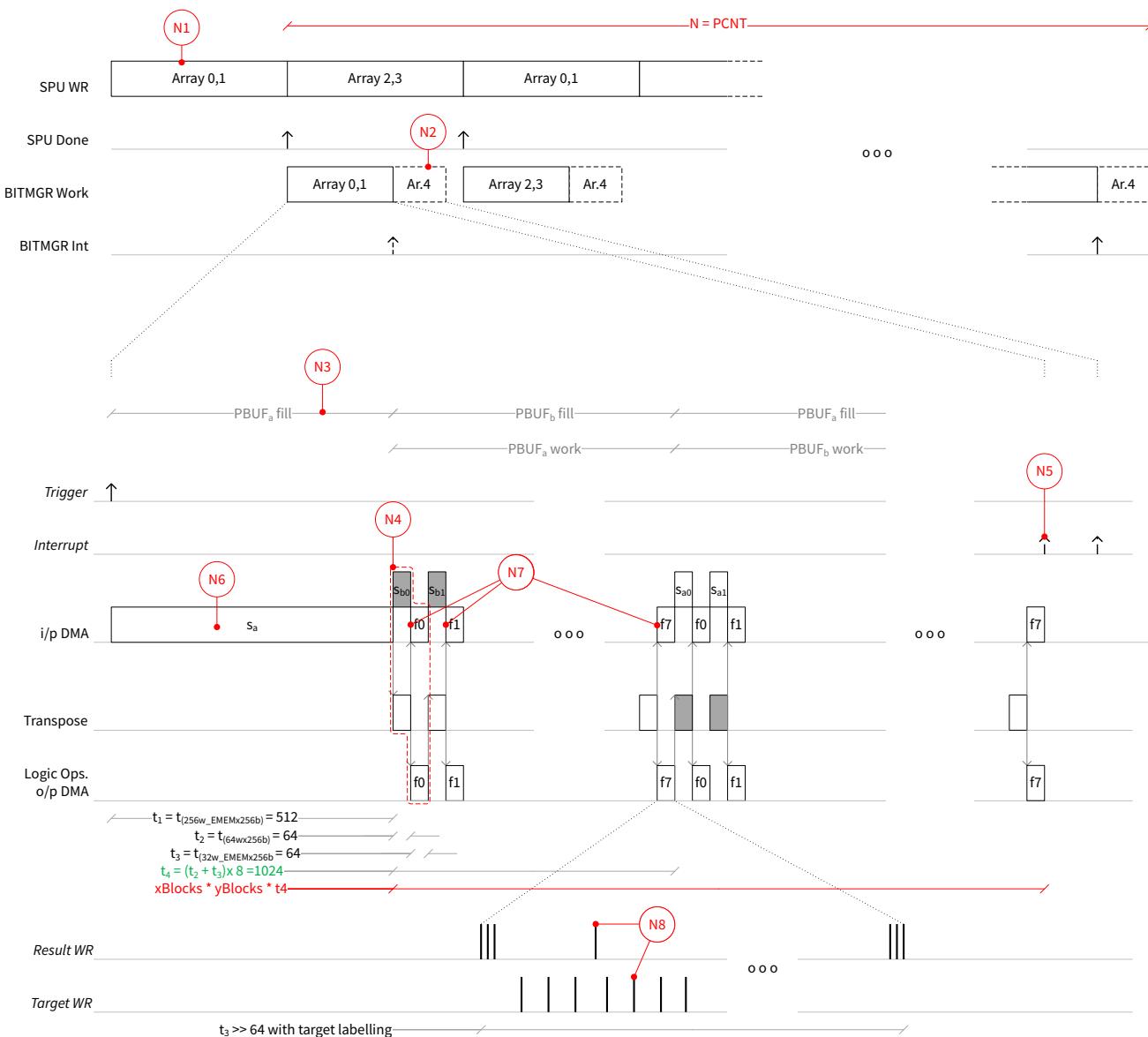
Event	Description	Note
Radar Memory Read Error	Error was seen on the RADAR memory Read port	STATCTRL.ERRMASK[0]
Radar Memory Write Error	Error was seen on the RADAR memory Write port	STATCTRL.ERRMASK[1]
Trigger overlap one	New trigger received when operation is in progress	STATCTRL.ERRMASK[2]

## Bit Manager (BITMGR)

### 21.4 Process pipeline

BITMGR interacts with SPU and rest of the system via triggers and interrupts and exchange data through EMEM buffers. The ideal case depicted in **Figure 275** as time dependencies on the pipeline is an indicative reference to optimize scheduling and mitigate resource conflicts. The pipeline illustrates one of the many scenario where array-operand size is greater than 256x256 and does not explicitly show differences due to chaining. This scenario was chosen as a good representation of the maximum throughput case. When both Mode0 and 1/2 are enabled each trigger executes the chained function and consequently interrupts are generated only towards the end of that operation. Refer **Chapter 21.3.4** for details. Note that target labelling is not deterministic, but dictated by the no of targets in the output instead.

Note: **Figure 275** is only an informational aid providing a thematic overview of the procedural time line. Due to variations on the workload imposed by a chosen configuration and other system level interactions, no claim is made to the exactness of the numbers depicted in the figure.



**Figure 275 Observable traffic pattern and process pipe**

---

**Bit Manager (BITMGR)**

Note: (1) SPU writes its results to memory, A ping-pong buffer in memory is available for continuous operation. (2) In optional chained operation mode. Figure only expands mode 0 (3) DMA activity with regards to Mode 0 execution (4) 256x256 blocks are processed in batches of 32 rows/columns. Entire operation on 32rows, 32 columns to produce 32 rows of result without target labelling depicted. (5) Interrupts may or may not be generated following every execution trigger depending on the trigger, run mode and interrupt configuration. (6) parts of one operand are read in continuously (7) parts of a second operand are read in burst and take priority over the read in [4]. 8 such bursts match the quantity read in [6] and also [4] (8) Optional target labelling operation and its impact on performance. Target labelling shares the write port, but exact sequence is variable depending on the number of identified targets and their density.

**Bit Manager (BITMGR)****21.5 Registers****Table 1035 Register Address Space - BITMGR**

Module	Base Address	End Address	Note
bitmgr_sfr	F9000000 <sub>H</sub>	F90007FF <sub>H</sub>	BBB Slave Interface to the BITMGR Special Function Registers

**Table 1036 Register Overview - BITMGR (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
CLC	Clock Control	000 <sub>H</sub>	17
MODID	Module Identification Register	004 <sub>H</sub>	18
ARYCFGz	Array Configuration z	008 <sub>H</sub> +z*4	19
BASEy	Base Address for Array y	010 <sub>H</sub> +y*4	20
CFG2D	Configuration of Dual Array Overlay (2D) Processing Mode	028 <sub>H</sub>	21
DAOBASE	DAO DMA Base Address	02C <sub>H</sub>	21
OFSTv	Address Offset v	030 <sub>H</sub> +v*4	23
CFG1D	Configuration of Sub-Array Overlay (1D) Processing Mode	04C <sub>H</sub>	23
SAOBASE	SAO DMA Base Address	050 <sub>H</sub>	25
TLBLB	Target Label Base	054 <sub>H</sub>	27
TLBLO	Target Label Offset	058 <sub>H</sub>	26
TLBLBASE	Target Label List Base Address	05C <sub>H</sub>	28
CTRL	Control register	060 <sub>H</sub>	28
TRIG	Trigger action register	064 <sub>H</sub>	30
REGCRC	Register CRC	068 <sub>H</sub>	32
TCNT	Target Count	06C <sub>H</sub>	32
STAT	Status and Reporting	070 <sub>H</sub>	33
STATCTRL	Status and Reporting Control	074 <sub>H</sub>	35
DATAd_CRC	Monitor CRC Register d	080 <sub>H</sub> +d*4	36
CTRLe_CRC	Monitor CRC Register e	180 <sub>H</sub> +e*4	36
CRC_CTRL_CRC	CRC of Control Monitor CRC Registers	200 <sub>H</sub>	37
CRC_MASK0	Control CRC Mask Register	204 <sub>H</sub>	38
CRC_MASK1	Spare Control CRC Mask Register	208 <sub>H</sub>	39
SMCTRL	Safety Mechanism Control Functions	20C <sub>H</sub>	39
SMSTAT	Safety Mechanism Status	210 <sub>H</sub>	41
SMUSER	Safety Mechanism Control Functions (User)	214 <sub>H</sub>	43
ACCEN0	Access Enable Register 0	218 <sub>H</sub>	44
ACCEN1	Access Enable Register 1	21C <sub>H</sub>	45
OCS	OCDS Control and Status	220 <sub>H</sub>	46
ODA	OCDS Debug Access Register	224 <sub>H</sub>	48
KRST0	Kernel Reset Register 0	228 <sub>H</sub>	48

## **Bit Manager (BITMGR)**

**Table 1036 Register Overview - BITMGR (ascending Offset Address) (cont'd)**

Short Name	Long Name	Offset Address	Page Number
KRST1	Kernel Reset Register 1	22C <sub>H</sub>	50
KRSTCLR	Kernel Reset Clear	230 <sub>H</sub>	51

### **21.5.1 Register Description**

## Clock Control

CLC

## Clock Control

(000<sub>H</sub>)

### **Reset Value: Table 1038**

The figure consists of two horizontal number lines. The top number line has labels at 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, and 16. The bottom number line has labels at 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, and 0. Both number lines have a central tick mark labeled '0'. The top line's '0' is positioned between 24 and 25, while the bottom line's '0' is positioned between 8 and 9.

Field	Bits	Type	Description
<b>DISR</b>	0	rw	<b>Disable Request</b> $0_B$ <b>Enable</b> , Request that the BITMGR clock tree be switched on $1_B$ <b>Disable</b> , Request that the BITMGR clock tree be switched off
<b>DISS</b>	1	rh	<b>Disable Status</b> This bit will be set to $1_B$ if the BITMGR kernel clock is disabled $0_B$ <b>Enable</b> , The BITMGR clock tree is switched on $1_B$ <b>Disabled</b> , The BITMGR clock tree is switched off
<b>EDIS</b>	3	rw	<b>Sleep Mode Enable Control</b> Reserved. This bit currently has no effect on the BITMGR. It should be kept at $0_B$ for future compatibility $0_B$ <b>Disabled</b> , Sleep Mode is Off $1_B$ <b>Enable</b> , Sleep Mode is On (no effect)
<b>0</b>	2, 31:4	r	<b>Reserved</b>

**Table 1037 Access Mode Restrictions of CLC sorted by descending priority**

<b>Mode Name</b>	<b>Access Mode</b>		<b>Description</b>
ENDINIT and Master enabled in ACCEN	rh	DISS	Write Accesses Permitted during Initialisation
	rw	DISR, EDIS	
Otherwise (default) (default)	r	DISR, EDIS	Default Access Mode (Bus Error on Write)
	rh	DISS	

**Bit Manager (BITMGR)****Table 1038 Reset Values of CLC**

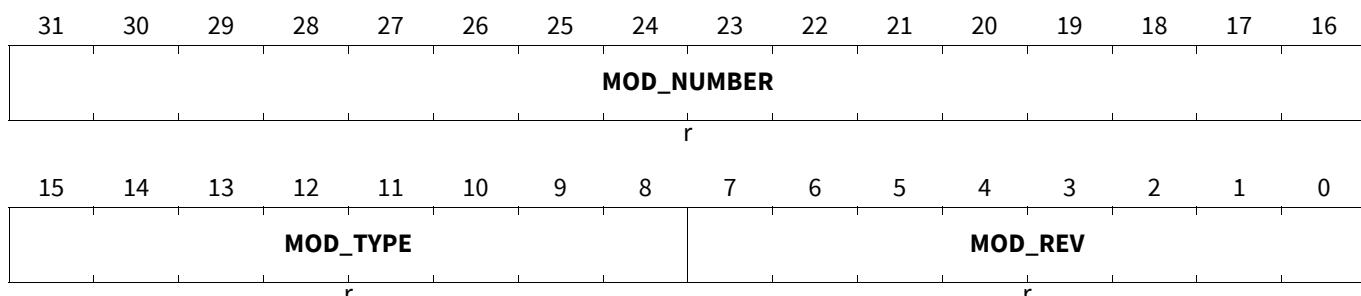
Reset Type	Reset Value	Note
Application Reset	0000 0003 <sub>H</sub>	Application Reset

**Module Identification Register**

This register contains information intended to enable identification of the module and the version of the module. The Revision Number (MOD\_REV) of the BITMGR is 01<sub>H</sub>.

**MODID**

**Module Identification Register** **(004<sub>H</sub>)** **Reset Value: Table 1040**



Field	Bits	Type	Description
<b>MOD_REV</b>	7:0	r	<b>Module Revision Number</b> This bit field defines the module revision number. The value of a module revision starts with 01H (first revision)
<b>MOD_TYPE</b>	15:8	r	<b>Module Type</b> Set to 0xC0 to indicate a 32 bit module
<b>MOD_NUMBE R</b>	31:16	r	<b>Module Number Value</b> Set to 0x00EC. This number is unique to the BITMGR.

**Table 1039 Access Mode Restrictions of MODID sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	MOD_NUMBER, MOD_REV, MOD_TYPE	Any write access will cause bus error
Otherwise (default)	r	MOD_NUMBER, MOD_REV, MOD_TYPE	Default Access Mode (Bus Error on Write)

**Table 1040 Reset Values of MODID**

Reset Type	Reset Value	Note
Application Reset	00EC C001 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	00EC C001 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Bit Manager (BITMGR)****Array Configuration z**

- Array Config 0 relates to array defined at Base 0,2 and Mode 0 result. Array defined at Base 1,3 inherit these values but interpret them with a coordinate exchange.
- Array Config 1 relates to array defined at Base 4,5 and Mode 1, 2 result .

**ARYCFGz (z=0-1)****Array Configuration z**(008<sub>H</sub>+z\*4)**Reset Value: Table 1042**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								YSIZE							
r								rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								XSIZE							
r								rw							

Field	Bits	Type	Description
<b>XSIZE</b>	9:0	rw	<b>Array 0,2,4,5 X Axis dimension</b> This field defines the X-dimension for Input Array 0,2,4,5 with index starting at 0. Array 1,3 has its co-ordinates exchanged by virtue of transposition and therefore inherits this value as its Y-dimension. Permitted values are 7 to 1023 both inclusive
<b>YSIZE</b>	25:16	rw	<b>Array 0,2,4,5 Y Axis dimension.</b> This field defines the Y-dimension for Input Array 0,2,4,5 with index starting at 0. Array 1,3 has its co-ordinates exchanged by virtue of transposition and therefore inherits this value as its X-dimension. Permitted values are 7 to 1023 both inclusive
<b>0</b>	15:10, 31:26	r	<b>Reserved</b>

**Table 1041 Access Mode Restrictions of ARYCFGz (z=0-1) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	XSIZE, YSIZE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	XSIZE, YSIZE	Default Access Mode (Bus Error on Write) (default)

**Table 1042 Reset Values of ARYCFGz (z=0-1)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Bit Manager (BITMGR)****Base Address for Array y**

- Base 0 and 1 are used as the base addresses of inputs to Mode 0
- Base 2 and 3 are counterparts to Base0 and 1 respectively in Ping-Pong buffer
- Base 4 is used as the base address for Mode1/2
- Base 5 is counterpart to Base4 to form a Ping-Pong buffer.

**BASEy (y=0-5)**

**Base Address for Array y** **(010<sub>H</sub>+y\*4)** **Reset Value: Table 1044**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							0								BASE
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE															
rw															

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Base Address for Array y</b> Base Address of the data to be read from Radar Memory. This is a 32 byte (256 bit) aligned address relative to the start address of the Radar Memory. Five LSBs need to be added to convert to a system address. This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.
<b>0</b>	31:19	r	<b>Reserved</b>

**Table 1043 Access Mode Restrictions of BASEy (y=0-5) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	BASE	Default Access Mode (Bus Error on Write)

**Table 1044 Reset Values of BASEy (y=0-5)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Bit Manager (BITMGR)

### Configuration of Dual Array Overlay (2D) Processing Mode

Configuration Options related to the Dual Array Overlay (2D) Processing (Mode 0)

#### CFG2D

#### Configuration of Dual Array Overlay [2D] Processing Mode(028<sub>H</sub>)

**Reset Value: Table 1046**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								0							
r				r				rw							

Field	Bits	Type	Description
<b>OPRTR</b>	3:0	rw	<b>Logical operation truth table</b> Defines the truth table for the 2 input, 1 output logical function used to combine elements from the two input arrays. The 4 bits provide a lookup for the output, where the 2 bit binary index to the table is generated by concatenating the input elements as {Array2 bit, Array 1 bit}.
<b>0</b>	9:4, 31:10	r	<b>RESERVED</b>

**Table 1045 Access Mode Restrictions of CFG2D sorted by descending priority**

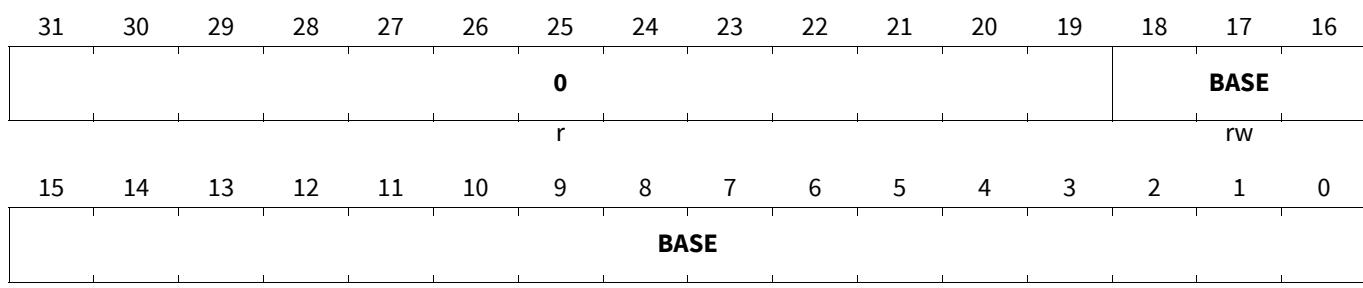
Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	OPRTR	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	OPRTR	Default Access Mode (Bus Error on Write)

**Table 1046 Reset Values of CFG2D**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### DAO DMA Base Address

Base Address for writing Dual Array Overlay processing(Mode0) results toRadar Memory. Result dimension is derived from ARYCFG0

**Bit Manager (BITMGR)****DAOBASE****DAO DMA Base Address****(02C<sub>H</sub>)****Reset Value: Table 1048**

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Base Address for Array</b> Base Address to be used when writing data to Radar Memory. This is a 32 byte (256 bit) aligned address relative to the start address of the Radar Memory. Five LSBs need to be added to convert to a system address. This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.
<b>0</b>	31:19	r	<b>Reserved</b>

**Table 1047 Access Mode Restrictions of DAOBASE sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	BASE	Default Access Mode (Bus Error on Write)

**Table 1048 Reset Values of DAOBASE**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Bit Manager (BITMGR)****Address Offset v****OFSTv (v=0-6)****Address Offset v**(030<sub>H</sub>+v\*4)**Reset Value: Table 1050**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>EN</b>								<b>0</b>							
rw								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						<b>0</b>									
						r									rw

Field	Bits	Type	Description
<b>ROW_OFST</b>	9:0	rw	<b>Row Offset for Array y</b> Index of the first row of a sub-array relative to row0 of its parent, Array4. A sub array placed at the defined offset will be included in sub array recombination, based on the state of EN. The maximum permissible offset is 1023 and the minimum offset is 1. If less than 7 sub-arrays are configured the valid offset definitions shall be placed contiguously starting from OFST0. Valid offsets must always be less than or equal to YSIZE .
<b>EN</b>	31	rw	<b>Enable this Offset</b> 0 <sub>B</sub> <b>NOT_VALID</b> , Exclude this row offset configuration 1 <sub>B</sub> <b>VALID</b> , Use this row offset configuration
<b>0</b>	30:10	r	<b>Reserved</b>

**Table 1049 Access Mode Restrictions of OFSTv (v=0-6) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	EN, ROW_OFST	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	EN, ROW_OFST	Default Access Mode (Bus Error on Write)

**Table 1050 Reset Values of OFSTv (v=0-6)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Configuration of Sub-Array Overlay (1D) Processing Mode**

Configuration Options for the 1D (Sub-Array Overlay or Row Combination) Processing (Mode1/2)

**Bit Manager (BITMGR)****CFG1D****Configuration of Sub-Array Overlay [1D] Processing Mode(04C<sub>H</sub>)****Reset Value: Table 1052**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>0</b>												<b>SAF</b>	<b>RPT</b>		
r												rw	rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>IS</b>	<b>0</b>	<b>ROWRPT</b>												<b>OPRTR</b>	
rw	r													rw	

Field	Bits	Type	Description
<b>OPRTR</b>	3:0	rw	<b>Logical Operator</b> Defines the truth table for the 2 input, 1 output logical function used to combine elements from sub-array rows. The 4 bits provide a lookup for the output, where the 2 bit binary index to the table is generated by concatenating the input elements as {Accumulator bit, Sub array bit}. The accumulator bit is initialised from Implicit Sub Array or OFST0. A single row result is obtained by applying the operator across the column elements in the row group reduced two bits at a time. Please refer the Mode1, 2 function description for further details.
<b>ROWRPT</b>	13:4	rw	<b>Row Repeat</b> Iteration count for Modes 1 or 2. For Mode1, the iteration defines the number of rows in a sub-array. For Mode2 the iteration define the number of sub-arrays of size RPT. This will therefore also influence the number of rows in the output array. The maximum is bounded by the YSIZE configuration, which shall not be exceeded. For further details please refer section on Execution Modes.
<b>IS</b>	15	rw	<b>Include Implicit Sub Array(IS) in Mode1/2 function.</b> Controls if the implicit sub-array, which is the one located at BASE, is included in computation for Mode1/2. When IS is included the operation starts from the BASE or row 0. When IS is excluded the operation starts from OFST0. When the implicit sub-array is excluded, software must configure OFST0 as the alternate start location. <b>0<sub>B</sub></b> <b>EXCLUDE</b> , Exclude the Implicit sub_array from computation. <b>1<sub>B</sub></b> <b>INCLUDE</b> , Include the Implicit sub_array in computation.
<b>RPT</b>	18:16	rw	<b>Row Read Repeat</b> Configures number of rows, N, to be combined in function Mode2, when RPT is configured to N-1. Allows N rows to be read from the input array, combined using an arbitrary logical operator and the combined result written back 'N' times to consecutive rows in the output array. Ignored when function Mode 2 is not enabled. Minimum and maximum values for N-1 are 1 and 1023 respectively.
<b>SAF</b>	19	rw	<b>Single Array Function</b> Enables Mode2 over Mode1 when set. <b>0<sub>B</sub></b> <b>MODE1</b> , Choose Mode 1 Function for Single array overlay <b>1<sub>B</sub></b> <b>MODE2</b> , Choose Mode 2 Function for Single array overlay

**Bit Manager (BITMGR)**

Field	Bits	Type	Description
0	14, 31:20	r	<b>RESERVED</b>

**Table 1051 Access Mode Restrictions of [CFG1D](#) sorted by descending priority**

Mode Name	Access Mode	Description
Master enabled in ACCEN	rw	IS, OPRTR, ROWRPT, RPT, SAF Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	IS, OPRTR, ROWRPT, RPT, SAF Default Access Mode (Bus Error on Write)

**Table 1052 Reset Values of [CFG1D](#)**

Reset Type	Reset Value	Note
Application Reset	0000 8000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 8000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**SAO DMA Base Address**

Base Address for writing Sub-Array Overlay processing (Mode1/2) results to Radar Memory. The X dimension of the result is derived from ARYCFG1, while the Ydimension is a function of ARYCFG1, ROWRPT and RPT.

**SAOBASE**

<b>SAO DMA Base Address</b>																<b>Reset Value: <a href="#">Table 1054</a></b>			
<b>(050<sub>H</sub>)</b>																			
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16																<b>0</b>			
																<b>BASE</b>			
																<b>r</b>			
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0																<b>BASE</b>			
																<b>rw</b>			

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Base Address for Array</b> Base Address to be used for writing data to Radar Memory. This is a 32 byte (256 bit) aligned address relative to the start address of the Radar Memory. Five LSBs need to be added to convert to a system address. This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.
0	31:19	r	<b>Reserved</b>

## Bit Manager (BITMGR)

**Table 1053 Access Mode Restrictions of SAOBASE sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	BASE	Default Access Mode (Bus Error on Write)

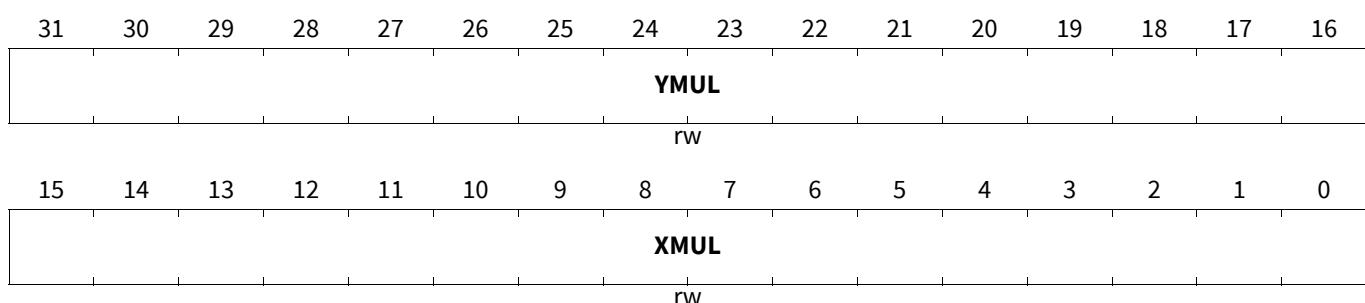
**Table 1054 Reset Values of SAOBASE**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Target Label Offset

### TLBLO

**Target Label Offset** **Reset Value: Table 1056**



Field	Bits	Type	Description
XMUL	15:0	rw	<b>X index multiplier</b> This value is multiplied by the x-index of a bin in the output array to remap the x-index to a relative byte offset address.
YMUL	31:16	rw	<b>Y index multiplier</b> This value is multiplied by the y-index of a bin in the output array to remap the y-index to a relative byte offset address.

**Table 1055 Access Mode Restrictions of TLBLO sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	XMUL, YMUL	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	XMUL, YMUL	Default Access Mode (Bus Error on Write)

## Bit Manager (BITMGR)

**Table 1056 Reset Values of TLBLO**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Target Label Base

Base Address to be used for Target Labelling Function. Added to the relative addresses computed by the labelling function to produce an absolute system addresses.

#### TLBLB

**Target Label Base** **(054<sub>H</sub>)** **Reset Value: Table 1058**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>BASE</b>															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>BASE</b>															
rw															
0															
r															

Field	Bits	Type	Description
<b>BASE</b>	31:5	rw	<b>Base Address</b>
<b>0</b>	4:0	r	<b>RESERVED</b>

**Table 1057 Access Mode Restrictions of TLBLB sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	BASE	Default Access Mode (Bus Error on Write)

**Table 1058 Reset Values of TLBLB**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Bit Manager (BITMGR)

### Target Label List Base Address

Base Address for writing the Target Label List to Radar Memory. The size of the buffer is fixed to 1024 EMEM locations

#### TLBLBASE

**Target Label List Base Address** **(05C<sub>H</sub>)** **Reset Value: Table 1060**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								BASE							
r															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE								rw							

Field	Bits	Type	Description
<b>BASE</b>	18:0	rw	<b>Base Address for Target List.</b> Base Address to be used when writing data to Radar Memory. This is a 32 byte (256 bit) aligned address relative to the start address of the Radar Memory. Five LSBs need to be added to convert to a system address. This field is sized to address a 16 MiB Radar Memory. Any MSBs not needed to address the memory in the product will be ignored. An overflow error (if configured) will be generated when the generated address exceeds the amount of physical Radar Memory available.
<b>0</b>	31:19	r	<b>Reserved</b>

**Table 1059 Access Mode Restrictions of TLBLBASE sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	BASE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	BASE	Default Access Mode (Bus Error on Write)

**Table 1060 Reset Values of TLBLBASE**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Control register

This register contains the options controlling the operation of the BITMGR. A control register write also serves as a mechanism to initialise internal state and counters.

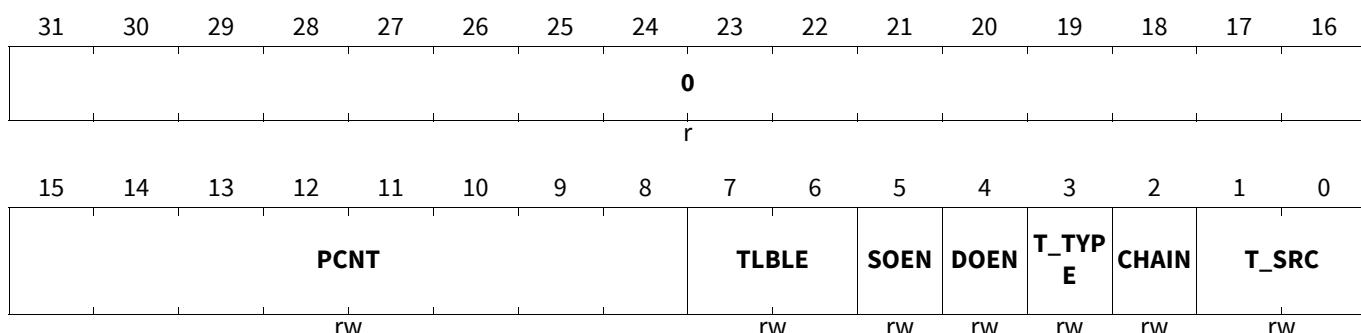
## Bit Manager (BITMGR)

### CTRL

#### Control register

(060<sub>H</sub>)

Reset Value: [Table 1062](#)



Field	Bits	Type	Description
T_SRC	1:0	rw	<b>Trigger source</b> Configures the Trigger source for the BITMGR. In normal operation, software shall not change any of the BITMGR registers when the value of this field is other than OFF. Trigger Action from TRIG.EVENT is conditioned on this field. 00 <sub>B</sub> <b>OFF</b> , Default after reset. Ignore triggers. 01 <sub>B</sub> <b>SWTRIG</b> , Software Trigger 10 <sub>B</sub> <b>SPU0</b> , Trigger on SPU0 DONE 11 <sub>B</sub> <b>SPU1</b> , Trigger on SPU1 DONE
CHAIN	2	rw	<b>Enable chaining of functions</b> Facilitates forwarding of the Dual array overlay function result to the Single Array overlay function, if both functions are enabled. Setting CHAIN alone does not remap the base address, but instead prevents the DAOBASE from incrementing when batch processing is enabled. User must configure all relevant BASEy addresses and ARYCFGz with compatible dimensions to link the chained operation.
T_TYPE	3	rw	<b>Single or batch processing</b> Specify the operating mode for BITMGR, single or batch mode. 0 <sub>B</sub> <b>ONESHOT</b> , Single Activation 1 <sub>B</sub> <b>CONTINUOUS</b> , Continuous Activation
DOEN	4	rw	<b>Dual Array Overlay Processing Enable</b> Mode0, Enable the dual-array overlay processing operation 0 <sub>B</sub> <b>DISABLE</b> , Disable Processing 1 <sub>B</sub> <b>ENABLE</b> , Enable Processing
SOEN	5	rw	<b>Sub-Array Overlay Processing Enable</b> Mode1/2, Enable the Sub-Array Overlay Processing Operation 0 <sub>B</sub> <b>DISABLE</b> , Disable Processing 1 <sub>B</sub> <b>ENABLE</b> , Enable Processing

## **Bit Manager (BITMGR)**

Field	Bits	Type	Description
<b>TLBLE</b>	7:6	rw	<p><b>Target label enable</b></p> <p>Enable the Target Labelling Function. The Base Address and Offset multiplier registers can be configured to produce relative or absolute addresses of the identified targets. Enabling this option will severely impact overall performance.</p> <p> <math>00_B</math> <b>DISABLE</b>, Disable Target Labelling  <math>01_B</math> <b>DAO</b>, Enable Target Labelling for Dual Array Overlay Results  <math>10_B</math> <b>SAO</b>, Enable Target Labelling for Sub-Array Overlay Results  <math>11_B</math> <b>RES</b>, Reserved       </p>
<b>PCNT</b>	15:8	rw	<p><b>Process Operation Count</b></p> <p>If CTRL.T_TYPE=CONTINUOUS then a counter is initialised to <math>0_D</math> when the CTRL register is written. This counter is incremented when a processing operation completes. The ATTENTION interrupt will be generated when the counter reaches the value of this field</p>
<b>0</b>	31:16	r	<b>Reserved</b>

**Table 1061 Access Mode Restrictions of CTRL sorted by descending priority**

<b>Mode Name</b>	<b>Access Mode</b>		<b>Description</b>
Master enabled in ACCEN	rw	CHAIN, DOEN, PCNT, SOEN, TLBLE, T_SRC, T_TYPE	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	CHAIN, DOEN, PCNT, SOEN, TLBLE, T_SRC, T_TYPE	Default Access Mode (Bus Error on Write)

**Table 1062** Reset Values of **CTRL**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
Application Reset	0000 0004 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Trigger action register

This register triggers BITMGR and reports status on triggered actionsBITMGR

TRIG

## Trigger action register

(064<sub>H</sub>)

### Reset Value: Table 1064

Memory Map Diagram:

- Address 31: **BUSY**
- Address 30: **T\_WAIT**
- Address 29: **PCNT\_STAT**
- Address 28: **r**
- Address 27: **0**
- Address 26: **0**
- Address 25: **0**
- Address 24: **0**
- Address 23: **0**
- Address 22: **0**
- Address 21: **0**
- Address 20: **0**
- Address 19: **0**
- Address 18: **0**
- Address 17: **0**
- Address 16: **0**
- Address 15: **0**
- Address 14: **0**
- Address 13: **0**
- Address 12: **0**
- Address 11: **0**
- Address 10: **0**
- Address 9: **0**
- Address 8: **0**
- Address 7: **0**
- Address 6: **0**
- Address 5: **0**
- Address 4: **0**
- Address 3: **0**
- Address 2: **0**
- Address 1: **0**
- Address 0: **0**

The PCNT\_STAT region is further divided into:

- EVENT**: Addresses 0 to 15.
- rwb**: Addresses 15 to 0.

**Bit Manager (BITMGR)**

Field	Bits	Type	Description
<b>EVENT</b>	0	rwh	<b>Trigger Event</b> Writing a 1 triggers BITMGR. If the T_SRC chosen in the CTRL register is set to SW trigger then BITMGR will be triggered as soon as the Register write completes, and write value for the bit is 1 If the T_SRC chosen in the CTRL register is set to HW trigger then BITMGR will be armed waiting for a trigger to be received on the chosen pin.
<b>PCNT_STAT</b>	15:8	rh	<b>Process Operation Count</b> Running status of PCNT when CTRL.T_TYPE=CONTINUOUS. The counter is initialised to 0 <sub>D</sub> when the CTRL register is written. This counter is incremented when a processing operation completes. The ATTENTION interrupt will be generated when the counter reaches the value of PCNT field
<b>T_WAIT</b>	30	rh	<b>Internal state</b> Internal status of BITMGR where it is waiting for a HW trigger to startprocessing data. 0 <sub>B</sub> <b>ARMED</b> , Waiting for HW trigger. 1 <sub>B</sub> <b>RUN</b> , Operation in progress
<b>BUSY</b>	31	rh	<b>Internal state</b> Internal status of BITMGR where an operation is in progress. In HW triggered mode, BUSY will be set even when waiting for a trigger, such as immediately after TRIG.EVENT write. 0 <sub>B</sub> <b>IDLE</b> , BitMgr has no outstanding operations. 1 <sub>B</sub> <b>BUSY</b> , Configuration complete, running following SW or HW trigger or, waiting HW trigger.
<b>0</b>	7:1, 29:16	r	<b>Reserved</b>

**Table 1063 Access Mode Restrictions of TRIG sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rh	BUSY, PCNT_STAT, T_WAIT	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rwh	EVENT	
Otherwise (default)	rh		Default Access Mode (Bus Error on Write)
(default)			

**Table 1064 Reset Values of TRIG**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

## Bit Manager (BITMGR)

### Register CRC

Expected CRC value for all registers from ARYCFG to REGCRC (excluding the TRIG register and CRC register itself)  
CRC checks are valid only between trigger event and CTRL.T\_SRC == "OFF"

### REGCRC

**Register CRC** **(068<sub>H</sub>)** **Reset Value: Table 1066**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRC															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC															
rw															

Field	Bits	Type	Description
CRC	31:0	rw	<b>CRC</b> The expected 32 bit CRC of all registers from ARYCFG to CTRL excluding the REGCRC register itself (which is replaced by 0x00000000)

**Table 1065 Access Mode Restrictions of REGCRC sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	CRC	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	CRC	Default Access Mode (Bus Error on Write)

**Table 1066 Reset Values of REGCRC**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Target Count

Stores a count of how many target entries (four targets per entry) have been written to the Target Label List

The count is initialised when the CTRL register is written.

The count will rollover. Bits 9 to 0 provide the current buffer index.

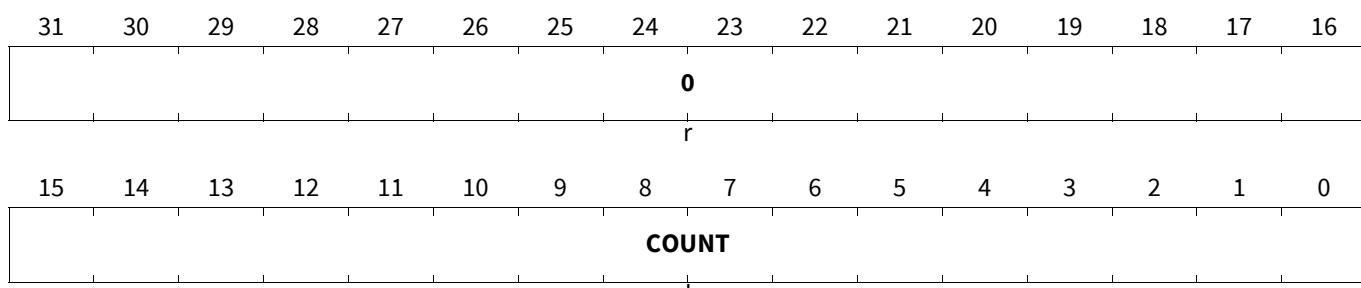
The higher order bits provide rollover status.

Ignoring the bit partition, the counter provides an estimate of targets identified since initialisation.

## Bit Manager (BITMGR)

### TCNT

**Target Count** (06C<sub>H</sub>) **Reset Value:** Table 1068



Field	Bits	Type	Description
COUNT	15:0	rh	<b>Target Count</b> This is a counter storing the number of entries written to the Target Label List since the last write to the CTRL register. The counter is read only and is cleared when the CTRL register is written.
0	31:16	r	<b>Reserved</b>

**Table 1067 Access Mode Restrictions of TCNT sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rh	COUNT	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	rh	COUNT	Default Access Mode (Bus Error on Write)

**Table 1068 Reset Values of TCNT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Status and Reporting

This register allows the interrupt trigger masks to be set and the triggers causing interrupts to be read. Refer to the Interrupts Section for bit allocation in the ERRMSK, INTMSK, ERRTRG and INTTRG fields

**Bit Manager (BITMGR)****STAT****Status and Reporting**(070<sub>H</sub>)Reset Value: [Table 1070](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															

**INTTRG**

r															
rh															

INTCL R	INTST S
w	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															

**ERRTRG**

r															
rh															

ERRCLR R	ERRST S
w	rwh

Field	Bits	Type	Description
<b>ERRSTS</b>	0	rwh	<b>Error Status</b> Set when an enabled error condition has triggered an interrupt
<b>ERRCLR</b>	1	w	<b>Error Clear</b> Set this bit to 1 while writing 0 to ERRSTS to clear the ERRSTS flag. Always reads as 0
<b>ERRTRG</b>	13:2	rh	<b>Error Trigger</b> The bits in this bitfield correspond to the bits in ERRMSK. Bits set will indicate which condition or conditions have triggered an interrupt. This field will be cleared when ERRSTS is cleared
<b>INTSTS</b>	16	rwh	<b>Interrupt Status</b> Set to 1 when an enabled interrupt condition (INTMSK) has triggered an interrupt
<b>INTCLR</b>	17	w	<b>Interrupt Clear</b> Set this bit to 1 while writing 0 to INTSTS to clear the Interrupt Status Flag. Always reads as 0
<b>INTTRG</b>	29:18	rh	<b>Interrupt Trigger</b> The bits in this bitfield correspond to the bits in INTMSK. Bits set will indicate which condition or conditions have triggered an interrupt. This field will be cleared when INTSTS is cleared
<b>0</b>	15:14, 31:30	r	<b>RESERVED</b>

**Table 1069 Access Mode Restrictions of STAT sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rh	ERRTRG, INTTRG	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rwh	ERRSTS, INTSTS	
	w	ERRCLR, INTCLR	
Otherwise (default) (default)	rX	ERRCLR, INTCLR	Default Access Mode (Bus Error on Write)
	rh	ERRSTS, ERRTRG, INTSTS, INTTRG	

## Bit Manager (BITMGR)

**Table 1070 Reset Values of STAT**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Status and Reporting Control

To be used by hardware to map and provide status on internal events sharing interrupt signals.

#### STATCTRL

**Status and Reporting Control** **(074<sub>H</sub>)** **Reset Value: Table 1072**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>0</b>														<b>0</b>	

r														r	
<b>INTMSK</b>															
<b>0</b>														<b>0</b>	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>ERRMSK</b>															

Field	Bits	Type	Description
ERRMSK	13:2	rw	<b>Error Mask</b> Mask field to enable interrupt on particular error conditions
INTMSK	29:18	rw	<b>Interrupt Mask</b> Mask Field to Enable/Disable Service Interrupt on particular events or conditions
0	1:0, 17:14, 31:30	r	<b>Reserved</b>

**Table 1071 Access Mode Restrictions of STATCTRL sorted by descending priority**

Mode Name	Access Mode	Description
Master enabled in ACCEN	rw	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	Default Access Mode (Bus Error on Write)

**Bit Manager (BITMGR)****Table 1072 Reset Values of STATCTRL**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

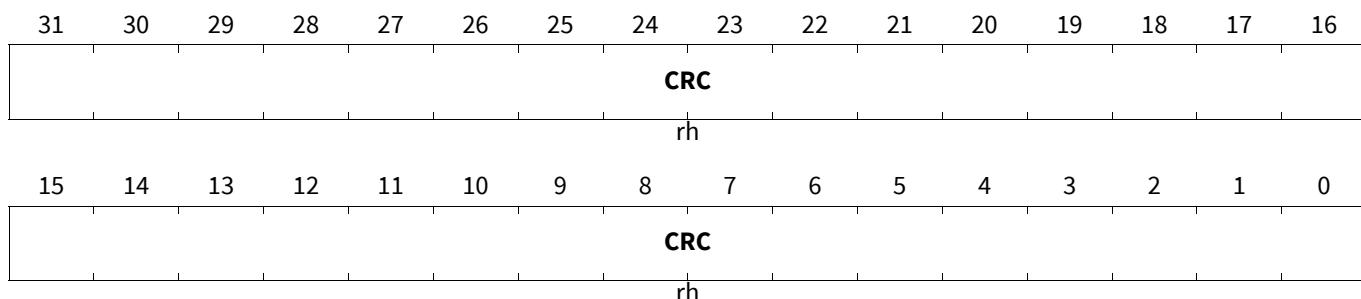
**Monitor CRC Register d**

CRC monitor data dependent signals within the BITMGR.

Allows read access to the output from one of the Monitor CRC Units

**DATAd\_CRC (d=0-63)**

**Monitor CRC Register d** (080<sub>H</sub>+d\*4) **Reset Value: Table 1074**



<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>CRC</b>	31:0	rh	<b>CRC</b> CRC value. Write 10 <sub>b</sub> to SMUSER.CINIT to clear

**Table 1073 Access Mode Restrictions of DATAd\_CRC (d=0-63) sorted by descending priority**

<b>Mode Name</b>	<b>Access Mode</b>		<b>Description</b>
Write Accesses	rh	CRC	Any write access will cause bus error
Otherwise (default)	rh	CRC	Default Access Mode (Bus Error on Write) (default)

**Table 1074 Reset Values of DATAd\_CRC (d=0-63)**

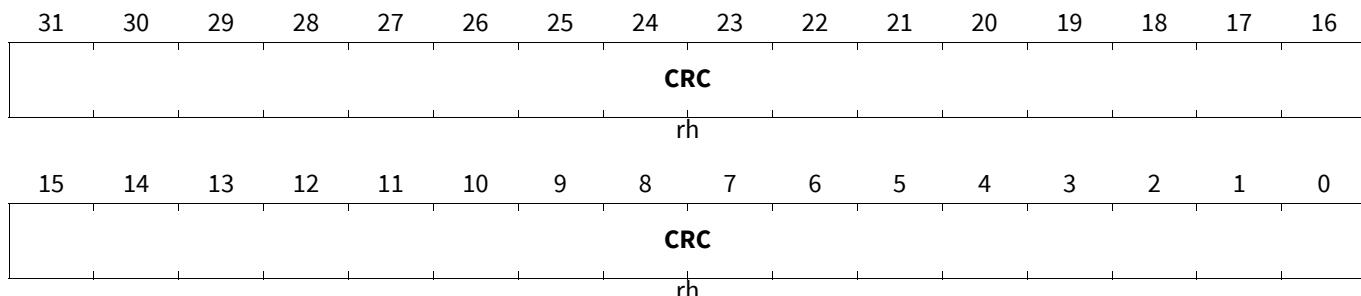
<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Monitor CRC Register e**

CRC Monitor for data independent control signals within the BITMGR.

**Bit Manager (BITMGR)**

Allows read access to the output from one of the Monitor CRC Units

**CTRLe\_CRC (e=0-31)****Monitor CRC Register e**(180<sub>H</sub>+e\*4)**Reset Value: Table 1076**

Field	Bits	Type	Description
CRC	31:0	rh	<b>CRC</b> CRC value. Write 10 <sub>b</sub> to SMUSER.CINIT to clear

**Table 1075 Access Mode Restrictions of CTRLe\_CRC (e=0-31) sorted by descending priority**

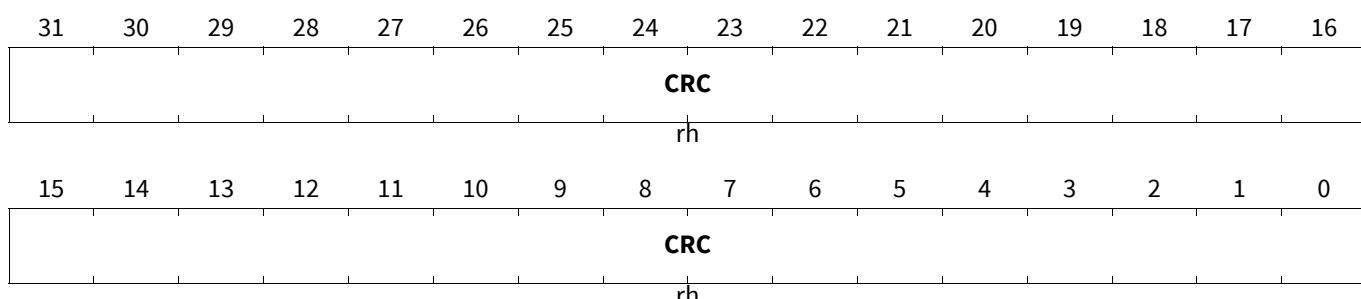
Mode Name	Access Mode		Description
Write Accesses	rh   CRC		Any write access will cause bus error
Otherwise (default)	rh   CRC		Default Access Mode (Bus Error on Write) (default)

**Table 1076 Reset Values of CTRLe\_CRC (e=0-31)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**CRC of Control Monitor CRC Registers**

Allows read access to the value of the overall CRC of the control CRC registers.

**CRC\_CTRL\_CRC****CRC of Control Monitor CRC Registers**(200<sub>H</sub>)**Reset Value: Table 1078**

## Bit Manager (BITMGR)

Field	Bits	Type	Description
CRC	31:0	rh	<b>CRC</b> CRC value. Write 10 <sub>b</sub> to SMUSER.CINIT to clear

**Table 1077 Access Mode Restrictions of [CRC\\_CTRL\\_CRC](#) sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	rh		CRC Any write access will cause bus error
Otherwise (default) (default)	rh		Default Access Mode (Bus Error on Write)

**Table 1078 Reset Values of [CRC\\_CTRL\\_CRC](#)**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Control CRC Mask Register

Include Control CRC registers in the overall CRC check.

#### **CRC\_MASK0**

#### **Control CRC Mask Register (204<sub>H</sub>) Reset Value: [Table 1080](#)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INC31	INC30	INC29	INC28	INC27	INC26	INC25	INC24	INC23	INC22	INC21	INC20	INC19	INC18	INC17	INC16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INC15	INC14	INC13	INC12	INC11	INC10	INC9	INC8	INC7	INC6	INC5	INC4	INC3	INC2	INC1	INC0
rw															

Field	Bits	Type	Description
INCI (i=0-31)	i	rw	<b>Include Control CRC Number i</b> Set this bit to include the corresponding Control CRC in an overall CRC calculation.

**Table 1079 Access Mode Restrictions of [CRC\\_MASK0](#) sorted by descending priority**

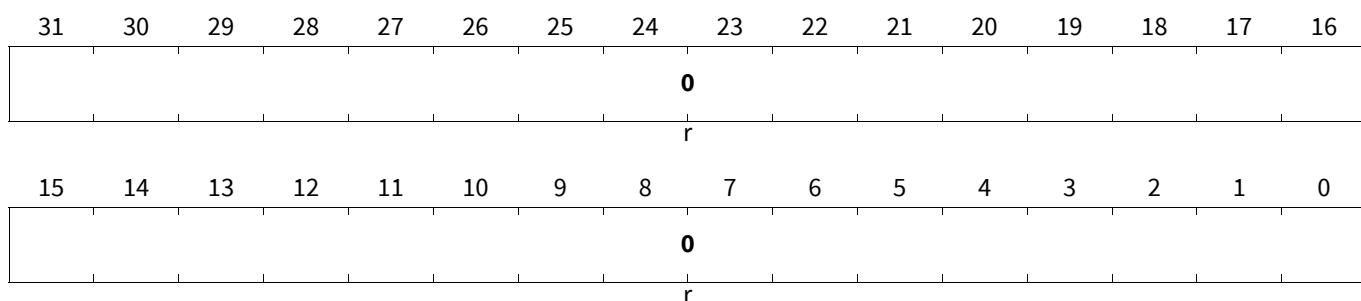
Mode Name	Access Mode		Description
Master enabled in ACCEN	rw	INCI (i=0-31)	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	INCI (i=0-31)	Default Access Mode (Bus Error on Write)

**Bit Manager (BITMGR)****Table 1080 Reset Values of CRC\_MASK0**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Spare Control CRC Mask Register**

RESERVED

**CRC\_MASK1****Spare Control CRC Mask Register (208<sub>H</sub>)****Reset Value: Table 1082**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
0	31:0	r	Reserved

**Table 1081 Access Mode Restrictions of CRC\_MASK1 sorted by descending priority**

<b>Mode Name</b>	<b>Access Mode</b>		<b>Description</b>
Write Accesses	-	See bit field definitions above	Any write access will cause bus error
Otherwise (default)	-	See bit field definitions above	Default Access Mode (Bus Error on Write) (default)

**Table 1082 Reset Values of CRC\_MASK1**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
Application Reset	0000 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Safety Mechanism Control Functions**

This register provides access to the control functions for safety mechanisms which are not normally expected to be modified as part of the normal configuration of the BITMGR.

**Bit Manager (BITMGR)****SMCTRL****Safety Mechanism Control Functions**(20C<sub>H</sub>)

Reset Value: Table 1084

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				RMTAERR		0				REGCRCEN		LREN			
r				rw		r				rw		rw			

Field	Bits	Type	Description
LREN	1:0	rw	<b>Logic Redundancy Enable</b> This bitfield allows control of the redundant logic and comparators monitoring the data processing operations. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit. 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>OFF</b> , Monitors are disabled 10 <sub>B</sub> <b>ON</b> , Monitors are enabled 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set
REGCRCEN	3:2	rw	<b>Register CRC Enable</b> Enable a Periodic CRC check of the register values against a known CRC stored in REGCRC.CRC. This checks registers at addresses between ARYCGF and CTRL inclusive but excluding the REGCRC register. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>OFF</b> , CRC is disabled 10 <sub>B</sub> <b>ON</b> , CRC is enabled 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set
RMTAERR	11:10	rw	<b>Radar Memory Access Address Error Enable</b> Enable an alarm to be generated if an access to the Radar Memory returns an error due to an invalid address. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>OFF</b> , Access Address Error is disabled 10 <sub>B</sub> <b>ON</b> , Access Address Error is enabled 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set
0	9:4, 31:12	r	<b>Reserved</b>

**Bit Manager (BITMGR)****Table 1083 Access Mode Restrictions of SMCTRL sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN and Safety ENDINIT	rw	LREN, REGCRCEN, RMTAERR	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default)	r	LREN, REGCRCEN, RMTAERR	Default Access Mode (Bus Error on Write) (default)

**Table 1084 Reset Values of SMCTRL**

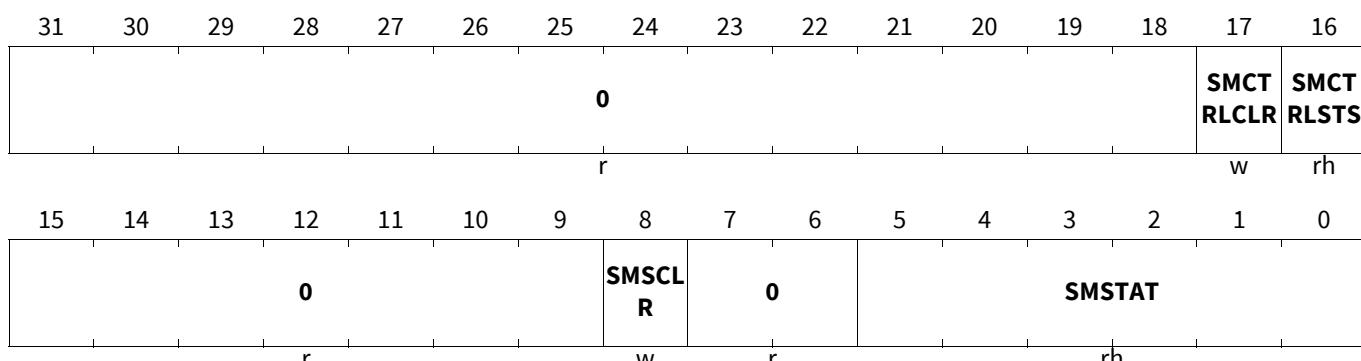
Reset Type	Reset Value	Note
Application Reset	0000 0405 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0000 0405 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**Safety Mechanism Status**

The BITMGR contains hardware safety mechanisms which monitor key areas of the BITMGR for correct functioning. In the event of an alarm being triggered by one of the safety mechanisms, this register can be read to determine which of the safety mechanisms has detected an error

**SMSTAT**

**Safety Mechanism Status** (210<sub>H</sub>) **Reset Value:** Table 1086



## Bit Manager (BITMGR)

Field	Bits	Type	Description
<b>SMSTAT</b>	5:0	rh	<p><b>Safety Mechanism Status</b></p> <p>Each bit will be set to one if the associated safety mechanism has detected a fault. The flags can be cleared by writing <math>1_b</math> to SMSTAT.SMSCLR. The values read from this bitfield are interpreted as follows (multiple failure conditions will cause the individual values to be summed):</p> <ul style="list-style-type: none"> <li><math>01_H</math> <b>LRCHK</b>, Mission Logic Failed Comparison with Redundant Logic Outputs</li> <li><math>02_H</math> <b>REGCRC</b>, Check of computed Register CRC against register value has failed</li> <li><math>04_H</math> <b>RDECC</b>, ECC Check Fail on Read from Radar Memory</li> <li><math>08_H</math> <b>RMCTRL</b>, Radar Memory Access Control Signals failed consistency check</li> <li><math>10_H</math> <b>RMTAERR</b>, Access to Radar Memory has resulted in an Error</li> </ul>
<b>SMSCLR</b>	8	w	<p><b>Clear Safety Mechanism Status</b></p> <p>Write <math>1_b</math> to this bit to clear the safety mechanism status flags. This bit will always read <math>0_b</math>.</p>
<b>SMCTRLSTS</b>	16	rh	<p><b>Safety Mechanism Control Status</b></p> <p>Safety Mechanism Control Logic Check has failed (inconsistent logic state)</p>
<b>SMCTRLCLR</b>	17	w	<p><b>Clear SMCTRL Status Flag</b></p> <p>Write <math>1_b</math> to this bit to clear the SMCTRL status flag. This bit will always read <math>0_b</math>.</p>
<b>0</b>	7:6, 15:9, 31:18	r	<b>Reserved</b>

Table 1085 Access Mode Restrictions of **SMSTAT** sorted by descending priority

Mode Name	Access Mode		Description
Master enabled in ACCEN	rh	SMCTRLSTS, SMSTAT	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	w	SMCTRLCLR, SMSCLR	
Otherwise (default) (default)	rX	SMCTRLCLR, SMSCLR	Default Access Mode (Bus Error on Write)
	rh	SMCTRLSTS, SMSTAT	

Table 1086 Reset Values of **SMSTAT**

Reset Type	Reset Value	Note
Application Reset	$0000\ 0000_H$	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	$0000\ 0000_H$	Kernel Reset (software controlled by KRST0-1 registers)

**Bit Manager (BITMGR)****Safety Mechanism Control Functions (User)**

This register provides access to the control functions for safety mechanisms not used as part of the normal configuration of the BITMGR. The various test bitfields defined in this register and used to test that the safety mechanisms can generate alarms are intended as a connectivity test only. It is therefore not necessary (or desirable) for the BITMGR to be running while these tests are run.

**SMUSER****Safety Mechanism Control Functions [User] (214<sub>H</sub>)****Reset Value: Table 1088**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RMTAERRTST	RDECCTST	RMCTRLTST	0				r	0			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				r	CINIT				rw						

Field	Bits	Type	Description
<b>CINIT</b>	1:0	rw	<b>Monitor CRC Unit Initialise</b> Write 10 <sub>b</sub> , to initialise Monitor CRC Units. The CRC monitors will be held in the initialisation status until 01 <sub>b</sub> is written. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit. 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>OFF</b> , No Error Injected 10 <sub>B</sub> <b>ON</b> , Error Injected 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set
<b>RMCTRLTST</b>	21:20	rw	<b>Test Radar Memory Control</b> Inject a deliberate error into the mechanism checking the consistency of the Radar Memory control signals to test the alarm generation. When set to 10 <sub>b</sub> , an alarm will be flagged until this field is written with 01 <sub>b</sub> . Writing an invalid value will set the SMSTAT.SMCTRLSTS bit. 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>OFF</b> , No Error Injected 10 <sub>B</sub> <b>ON</b> , Error Injected 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set
<b>RDECCTST</b>	23:22	rw	<b>Test EMEM Read Data ECC</b> Inject a deliberate error into the mechanism checking the ECC of data read from the Radar Memory to test the alarm generation. When set to 10b, all ECC checks should fail until this field is written with 01b. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit. 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>OFF</b> , No Error Injected 10 <sub>B</sub> <b>ON</b> , Error Injected 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set

## Bit Manager (BITMGR)

Field	Bits	Type	Description
RMTAERRTST	25:24	rw	<b>Test Radar Memory Access Address Error</b> Inject a deliberate error into the mechanism checking for an address error condition being flagged for a access to Radar Memory to test the alarm generation. When set to 10b, all accesses to Radar Memory should result in a alarm until this field is written with 01b. Writing an invalid value will set the SMSTAT.SMCTRLSTS bit. 00 <sub>B</sub> <b>ERR0</b> , Invalid. SMCTRLSTS will be set 01 <sub>B</sub> <b>OFF</b> , No Error Injected 10 <sub>B</sub> <b>ON</b> , Error Injected 11 <sub>B</sub> <b>ERR3</b> , Invalid. SMCTRLSTS will be set
0	19:2, 31:26	r	<b>Reserved</b>

**Table 1087 Access Mode Restrictions of SMUSER sorted by descending priority**

Mode Name	Access Mode			Description
Master enabled in ACCEN	rw	CINIT, RDECCTST, RMCTRLTST, RMTAERRTST		Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	r	CINIT, RDECCTST, RMCTRLTST, RMTAERRTST		Default Access Mode (Bus Error on Write)

**Table 1088 Reset Values of SMUSER**

Reset Type	Reset Value	Note
Application Reset	0150 0000 <sub>H</sub>	Application Reset
Kernel Reset (software controlled by KRST0-1 registers)	0150 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Access Enable Register 0

The Access Enable Register 0 controls access for writes to registers and configuration memory using the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BITMGR is prepared for a 6 bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ..., EN31 -> TAG ID 011111B.

#### ACCEN0

#### Access Enable Register 0 (218<sub>H</sub>) Reset Value: [Table 1090](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

**Bit Manager (BITMGR)**

Field	Bits	Type	Description
<b>ENx (x=0-31)</b>	x	rw	<p><b>Access Enable for Master TAG ID x</b></p> <p>This bit enables access to the BITMGR register addresses for transactions with the Master TAG ID x</p> <p><math>0_B</math> <b>RO</b>, Write access will terminate without error but will not be executed.</p> <p><math>1_B</math> <b>RW</b>, Write and read accesses will be executed</p>

**Table 1089 Access Mode Restrictions of ACCENO sorted by descending priority**

Mode Name	Access Mode		Description
Safety ENDINIT	rw	ENx (x=0-31)	Write access permitted during safety initialisation only. Write Accesses at other time errored
Otherwise (default) (default)	r	ENx (x=0-31)	Default Access Mode (Bus Error on Write)

**Table 1090 Reset Values of ACCENO**

Reset Type	Reset Value	Note
Application Reset	FFFF FFFF <sub>H</sub>	Application Reset

**Access Enable Register 1**

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). These tags are not used in this system and so no programmable bits are provided.

**ACCEN1**

(21C <sub>H</sub> )																Reset Value: <a href="#">Table 1092</a>			
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16																0			
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0																0			

Field	Bits	Type	Description
<b>0</b>	31:0	r	<p><b>Reserved</b></p> <p>Read as 0, should be written with 0.</p>

## Bit Manager (BITMGR)

**Table 1091 Access Mode Restrictions of ACCEN1 sorted by descending priority**

Mode Name	Access Mode		Description
Safety ENDINIT	-	See bit field definitions above	Write access permitted during safety initialisation only. Write Accesses at other time errored
Otherwise (default) (default)	-	See bit field definitions above	Default Access Mode (Writes silently ignored)

**Table 1092 Reset Values of ACCEN1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset

### OCDS Control and Status

The OCDS Control and Status (OCS) register controls the module's behaviour in suspend mode (used for debugging).

The register can only be written when the OCDS is enabled. While OCDS is disabled, the OCDS suspend control is ineffective .

### OCS

#### OCDS Control and Status (220<sub>H</sub>) Reset Value: Table 1094

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		<b>SUSST A</b>	<b>SUS_P</b>		<b>SUS</b>							<b>0</b>			
r	rh	w		rw								r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						<b>0</b>						<b>TG_P</b>	<b>TGB</b>	<b>TGS</b>	
						r						rw	rw	rw	

Field	Bits	Type	Description
<b>TGS</b>	1:0	rw	<b>Trigger Bus Select</b> Select which of the two possible trigger busses are connected to the output 00 <sub>B</sub> <b>OFF</b> , No Trigger Set Output 01 <sub>B</sub> <b>SETA</b> , Set A Output Triggers are driven onto the output 10 <sub>B</sub> <b>SETB</b> , Set B Output Triggers are driven onto the output 11 <sub>B</sub> <b>RES</b> , Reserved Do Not Use
<b>TGB</b>	2	rw	<b>OTGB0/1 Bus Select</b> 0 <sub>B</sub> <b>OTGB0</b> , Trigger Set is Output on OTGB0 1 <sub>B</sub> <b>OTGB1</b> , OTGB1 Trigger Bus is Output on OTGB1

### Bit Manager (BITMGR)

Field	Bits	Type	Description
<b>TG_P</b>	3	rw	<b>TGS, TGB Write Protection</b> TGS and TGB are written only when TG_P is 1, otherwise unchanged. TG_P always reads 0
<b>SUS</b>	27:24	rw	<b>Suspend</b> Enable Suspend of the BITMGR on Debug System Break. All values for this field not explicitly enumerated will cause a soft suspend of the BITMGR on break $0_H$ <b>RUN</b> , Break has no effect on BITMGR operation $1_H$ <b>HARD</b> , Break causes hard suspend of BITMGR
<b>SUS_P</b>	28	w	<b>Suspend Protect</b> SUS bitfield can only be updated if this bit is set to 1 for the write access
<b>SUSSTA</b>	29	rh	<b>Suspend Status</b> This field will be set to one if the BITMGR is suspended
<b>0</b>	23:4, 31:30	r	<b>Reserved</b>

**Table 1093 Access Mode Restrictions of OCS sorted by descending priority**

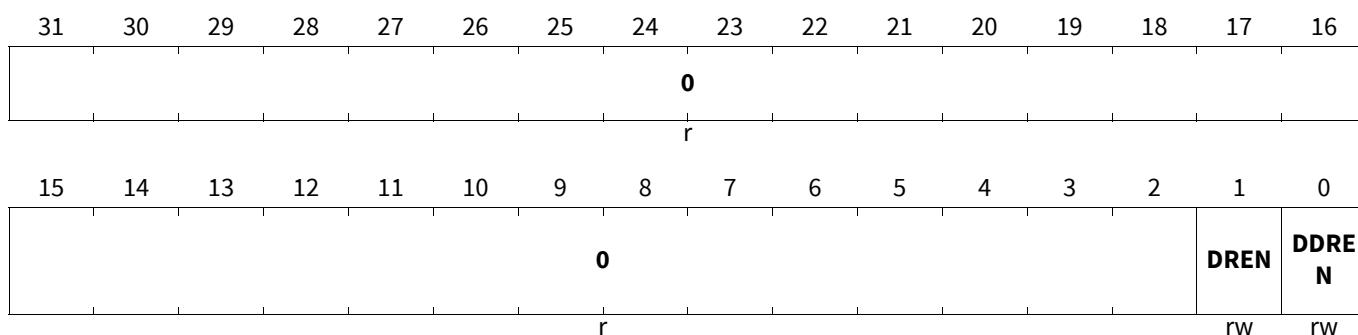
Mode Name	Access Mode		Description
Master enabled in ACCEN and Supervisor Mode and OCDS Is Enabled	rh	SUSSTA	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	TGB, TGS, TG_P	
	w	SUS_P	
Master enabled in ACCEN and OCDS Is Enabled and Supervisor Mode and write 1 to <b>SUS_P</b>	rw	SUS	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Supervisor Mode	r	SUS, TGB, TGS, TG_P	Read Access Permitted for Supervisor Mode Accesses
	rX	SUS_P	
	rh	SUSSTA	
Otherwise (default) (default)	r	SUS, TGB, TGS, TG_P	Default Access Mode (Bus Error on Write)
	rX	SUS_P	
	rh	SUSSTA	

**Table 1094 Reset Values of OCS**

Reset Type	Reset Value	Note
Power On Reset	0000 0000 <sub>H</sub>	"Power on Reset"

### OCDS Debug Access Register

The BITMGR currently has no registers affected by destructive read. This register is reserved for future extension.

**Bit Manager (BITMGR)****ODA****OCDS Debug Access Register**(224<sub>H</sub>)Reset Value: [Table 1096](#)

Field	Bits	Type	Description
<b>DDREN</b>	0	rw	<b>Destructive Debug Read Enable</b> If set, reads by the OCDS master to bits which would normally change state on read, do not cause the state to change
<b>DREN</b>	1	rw	<b>Destructive Read Enable</b> If set, reads to bits which would normally change state on read, do not cause the state to change
<b>0</b>	31:2	r	<b>Reserved</b>

**Table 1095 Access Mode Restrictions of ODA sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN and Supervisor Mode	rw	DDREN, DREN	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Supervisor Mode	r	DDREN, DREN	Read Access Permitted for Supervisor Mode Accesses
Otherwise (default) (default)	r	DDREN, DREN	Default Access Mode (Bus Error on Write)

**Table 1096 Reset Values of ODA**

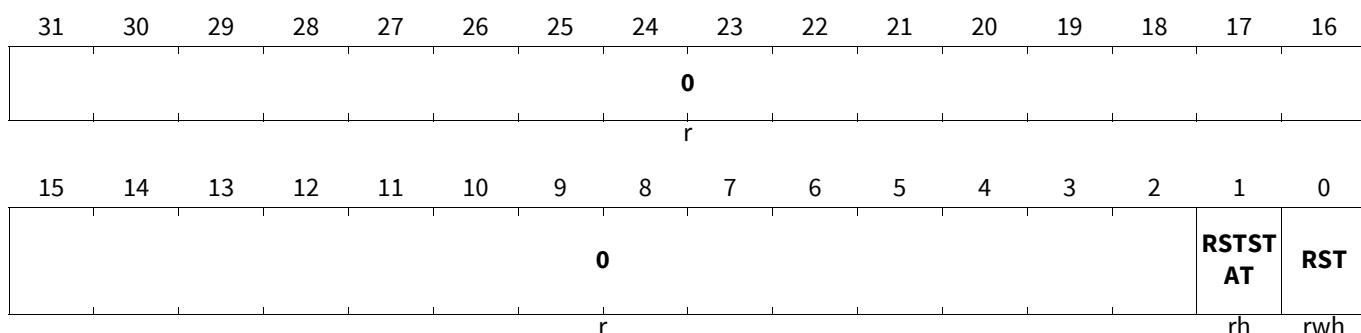
Reset Type	Reset Value	Note
Power On Reset	0000 0000 <sub>H</sub>	"Power on Reset"

**Kernel Reset Register 0**

The KRST0, KRST1 and KRSTCLR registers are used to initiate and monitor a reset of the BITMGR kernel logic. The kernel reset is equivalent to a full application reset, only the registers implemented in the bus interface are not affected. All registers affected by kernel reset have an appropriate entry in the "reset values" table

Note, that if a debugger is connected, an application reset of the device will have the same effect on the KRST0 and KRST1 registers as a kernel reset. If KRST0.RST or KRST1.RST are set, they will be cleared and the KRST0.RSTSTAT bit will be set.

Note, this register is an exception to the general use of ENDINIT to protect register related to Kernel Reset. This register is protected only by ACCEN.

**Bit Manager (BITMGR)****KRST0****Kernel Reset Register 0**(228<sub>H</sub>)Reset Value: [Table 1098](#)

Field	Bits	Type	Description
<b>RST</b>	0	rwh	<b>Kernel Reset</b> This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to 0B) after the kernel reset is executed 0 <sub>B</sub> <b>Run</b> , Normal Operation 1 <sub>B</sub> <b>Reset</b> , Request a kernel reset of the BITMGR This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to 0B) after the kernel reset is executed
<b>RSTSTAT</b>	1	rh	<b>Kernel Reset Status</b> This bit indicates whether a kernel reset was executed or not. This bit is set after the execution of a kernel reset in the same clock cycle both reset bits are cleared (KRST0.RST and KRST1.RST). 0 <sub>B</sub> <b>Run</b> , No reset has occurred 1 <sub>B</sub> <b>Reset</b> , A kernel reset was executed
<b>0</b>	31:2	r	<b>Reserved</b> Will always read as 0 <sub>B</sub> . Should be written with 0 <sub>B</sub>

**Table 1097 Access Mode Restrictions of KRST0 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rh	RSTSTAT	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rwh	RST	
Otherwise (default) (default)	rh	RST, RSTSTAT	Default Access Mode (Bus Error on Write)

**Table 1098 Reset Values of KRST0**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset

## Bit Manager (BITMGR)

### Kernel Reset Register 1

The KRST0, KRST1 and KRSTCLR registers are used to initiate and monitor a reset of the BITMGR kernel logic. The kernel reset is equivalent to a full application reset, only the registers implemented in the bus interface are not affected. All registers affected by kernel reset have an appropriate entry in the "reset values" table

Note, that if a debugger is connected, an application reset of the device will have the same effect on the KRST0 and KRST1 registers as a kernel reset. If KRST0.RST or KRST1.RST are set, they will be cleared and the KRST0.RSTSTAT bit will be set.

Note, this register is an exception to the general use of ENDINIT to protect register related to Kernel Reset. This register is protected only by ACCEN.

### KRST1

**Kernel Reset Register 1** **Reset Value: Table 1100**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															rwh

Field	Bits	Type	Description
RST	0	rwh	<b>Kernel Reset</b> This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to 0B) after the kernel reset is executed 0 <sub>B</sub> <b>Run</b> , Normal Operation 1 <sub>B</sub> <b>Reset</b> , Request a kernel reset of the BITMGR This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to 0B) after the kernel reset is executed
0	31:1	r	<b>Reserved</b> Will always read as 0 <sub>B</sub> . Should be written with 0 <sub>B</sub>

**Table 1099 Access Mode Restrictions of KRST1 sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	rwh	RST	
Otherwise (default)	rh	RST	

**Table 1100 Reset Values of KRST1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset

## Bit Manager (BITMGR)

### Kernel Reset Clear

The KRST0, KRST1 and KRSTCLR registers are used to initiate and monitor a reset of the BITMGR kernel logic. The kernel reset is equivalent to a full application reset, only the registers implemented in the bus interface are not affected. All registers affected by kernel reset have an appropriate entry in the "reset values" table

Note, that if a debugger is connected, an application reset of the device will have the same effect on the KRST0 and KRST1 registers as a kernel reset. If KRST0.RST or KRST1.RST are set, they will be cleared and the KRST0.RSTSTAT bit will be set.

Note, this register is an exception to the general use of ENDINIT to protect register related to Kernel Reset. This register is protected only by ACCEN.

### KRSTCLR

**Kernel Reset Clear** (230<sub>H</sub>) **Reset Value:** Table 1102

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							0								
								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							0								
								r							w

Field	Bits	Type	Description
CLR	0	w	<b>Kernel Reset Status Clear</b> Write 1 <sub>B</sub> to clear the KRST0.RSTSTAT bit. Always reads as 0 <sub>B</sub> . 0 <sub>B</sub> <b>Run</b> , No Action 1 <sub>B</sub> <b>Clear</b> , Clear Kernel Reset Status (KRST0.RSTSTAT)
0	31:1	r	<b>Reserved</b> Will always read as 0 <sub>B</sub> . Should be written with 0 <sub>B</sub>

**Table 1101 Access Mode Restrictions of KRSTCLR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	w	CLR	Write Access for permitted masters only, writes for non-permitted masters cause bus error
Otherwise (default) (default)	rX	CLR	Default Access Mode (Bus Error on Write)

**Table 1102 Reset Values of KRSTCLR**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	Application Reset

## Bit Manager (BITMGR)

### 21.6 Debug

For purposes of debug, it is possible to stop execution at well defined functional handshakes by Software or using the system wide OCDS suspend functionality to break an autonomous cycle.

For availability reasons, the command to stop execution requires to write 3 bits,  $111_B$ , defined as STOP, to the CTRL.MODE bitfield

When such finer control is exercised by the debugger, operation cannot resume after the break. Once the OCDS suspend has been removed, the BITMGR should be initialised using kernel reset.

Two types of suspend are available

- Hard Suspend: this stops the kernel clock of the BITMGR immediately. Registers can still be read but should not be written. EMEM control signals may suspend in the active state.
- Soft Suspend: Operation is suspended when the control logic reaches a stable state and the interfaces are transitioned to a benign state. Suspend will be acknowledged as soon all outstanding writes to Radar Memory complete.

#### 21.6.1 Trace

BITMGR will not use the trace port as data is easily visible in memory.

#### 21.6.2 Debugger events

The following event triggers are available as hardware events for the debugger via the configuration options in the OCS register.

**Table 1103 OTGB Hardware Triggers (Set A)**

0	Started a unit of work
1	Completed a unit of work
2	Completed Mode0
3	Started Mode1/2
4	Label list half water mark reached
5	Label list full water mark reached
6	Implementation
7	Implementation
8	Implementation
9	Implementation
10	
11	
12	
13	
14	
15	

**Table 1104 OTGB Hardware Triggers (Set B)**

0	Read error signaled by Radar Memory
1	Out of range address generated

## Bit Manager (BITMGR)

**Table 1104 OTGB Hardware Triggers (Set B) (cont'd)**

2	Insufficient process time - new trigger received when previous unit of work was in progress.
3	Implementation specific
4	Implementation specific
5	Implementation specific
6	Implementation specific
7	
8	
9	
10	
11	
12	
13	
14	
15	

## 21.7 Safety Measures

The BITMGR has the following safety mechanisms to assist in its use in safety related applications

### 21.7.1 Hardware Safety Mechanisms

The BITMGR has the following autonomous hardware safety mechanisms which report failures via SMU alarms

#### 21.7.1.1 Functional logic redundancy

The internal functional logic apart from RAMs within BITMGR is implemented with a fully redundant copy. This redundant copy is neither inverted nor delayed. At runtime, a checker checks for discrepancies between the copies and generates an SMU alarm in case of a mismatch.

#### 21.7.1.2 Register CRC

The register CRC uses a 32 bit ethernet polynomial which assumes that the register contents are converted to an LSB first sequential data stream. The data stream will be generated using the registers from **ARYCFG<sub>z</sub> (z=0-1)** to **CTRL** inclusive. Unused register addresses shall be replaced by 0000 0000<sub>H</sub> in the datastream.

The polynomial shall run periodically once enabled and the CRC generated from the register contents compared with the value stored in the **REGCRC.CRC** bitfield. If the comparison fails, then an SMU alarm will be generated.

The Register CRC is enabled by writing 10<sub>B</sub> to the **SMCTRL.REGCRCEN** bitfield and disabled by writing 01<sub>B</sub>.

Control and comparison logic of the safety mechanism is replicated using negated logic. Any difference in behaviour between the two sets of logic will result in setting of the **SMSTAT.SMCTRLSTS** bit.

The alarm can be tested by writing an invalid CRC to **REGCRC.CRC**.

If this condition has triggered an alarm, bit 1 of the **SMSTAT.SMSTAT** bitfield will be set to 1<sub>B</sub>.

#### 21.7.1.3 Radar Memory Control Signal Redundancy

All control signals controlling accesses to Radar Memory are duplicated. The duplicate signals are set to the negated value of the primary control signal. In the event of the two signals becoming inconsistent, it can be

## Bit Manager (BITMGR)

assumed that an error has occurred. An inconsistent state is detected by comparators and will cause an SMU alarm.

This check is permanently enabled. If this condition has triggered an alarm<sup>1)</sup>, bit 4 of the **SMSTAT**.SMSTAT bitfield will be set to 1<sub>B</sub>.

### 21.7.1.4 Radar Memory Tile Access Error

Radar Memory tiles are enabled for BITMGR accesses by configuring the tile control mechanism in the EMEM. An BITMGR access can be directed to a blocked tile either by a configuration error or a fault in the BITMGR logic. In the event of the EMEM reporting that an access has failed because it has been mapped to a blocked tile, this mechanism can be used to trigger an SMU alarm.

The alarm is enabled by writing 10<sub>B</sub> to the **SMCTRL**.RMTAERR bitfield and disabled by writing 01<sub>B</sub>. Writing either of the other two possible values will result in the SMSTAT.SMCTRLSTS bit being set.

If this condition has triggered an alarm, bit 5 of the SMSTAT.SMSTAT bitfield will be set to 1<sub>B</sub>.

This alarm condition can be tested by writing 10<sub>B</sub> to the SMUSER.RMTAERRTST bitfield.

### 21.7.1.5 Radar Memory Read Data ECC

Read data returned for a Radar Memory access by the BITMGR is protected by an ECC computed in the EMEM from the address presented to the RAM and the data output from the RAM. This is checked by the BITMGR to ensure that the data is correct and has been read from the expected RAM location. A failed check will be signalled via an SMU alarm

The alarm is always enabled.

If this condition has triggered an alarm, bit 3 of the SMSTAT.SMSTAT bitfield will be set to 1<sub>B</sub>.

This alarm condition can be tested by writing 10<sub>B</sub> to the SMUSER.RDECCTST bitfield.

### 21.7.1.6 Private buffers based on RAM

All RAMs used to build private buffers are instances of safety qualified library components. Therefore no additional mechanism are required to be implemented to instance such module. In addition, the SSH interfaces to functional logic are fully compliant.

### 21.7.1.7 Access Enable

Write accesses to the registers are controlled by the Access Enable mechanism. This is configured using the ACCEN0/ACCEN1 registers and uses the master tag used to identify the originating master of each write transaction to permit or deny access to modify the contents of the register or memory.

## 21.7.2 Software Based Safety Mechanisms

The BITMGR supports the following safety mechanisms which require support from software. Failures are to be reported by the software rather than via the SMU

## 21.7.3 Hardware Functionality Supporting Software Safety Mechanisms

The following hardware functionality exists within the BITMGR to increase the effectiveness of the Software Based Safety Mechanisms

1) The Error flag returned by the EMEM for an access address error is also a redundant signal but, as this is already a monitor rather than a control, an inconsistency in this pair causes the SMSTAT.SMCTRLSTS bit to set and does not generate an alarm

## Bit Manager (BITMGR)

### 21.7.3.1 Monitor CRC Units

These are place holders for future use.

The BITMGR may implement CRC units used to provide a method of checking the activity on key internal interfaces. The CRC results are made available to software via registers.

The CRCs shall be separated into two blocks in the registers space.

- CRCs which are independent of the data being processed. These are enabled using **SMCTRL**.CTRLCRCEN
- CRCS which are dependent on the data being processed. These are enabled using **SMCTRL**.DATACRCEN

The first can be used for providing some detection of soft errors occurring during execution as well as during software based self test. The second can only be used while processing known data during software based self test.

The CRC units are mapped into the BITMGR register space as read only registers.

CRC functions shall be disabled if the corresponding mask is set or globally if the SMCTRL.CTRLCRCEN or SMCTRL.DATACRCEN bit is set.

The CRC Units have an associated control field (**SMUSER**.CINIT). The CRC units are initialised to the reset value by writing  $10_B$  to **SMUSER**.CINIT.

**Note:** *With full logic redundancy, it is expected that the CRC registers will be defunct in which case they are treated as reserved. As these are read only registers they do not interfere with the workings of mission mode logic.*

### 21.7.3.2 Redundant Control Logic

All control and status state bits (flip-flops) implementing safety mechanism functionality are duplicated. The duplicate flip-flops are used to store the negated value of the bit. This can be observed directly in the **SMCTRL** and **SMSTAT** registers. In the event of the state of these bits becoming inconsistent, it can be assumed that either a programming or soft error has occurred. An inconsistent state is detected by comparators and can be observed by polling the **SMSTAT**.SMCTRLSTS bit.

### 21.7.4 SMU events

The following table list the events that can trigger an SMU alarm. See [Section 21.7.1](#) for a description of the hardware safety mechanisms.

**Table 1105 SMU Events**

Event	Definition	Source	Comments
uncorrectable ECC error in Radar memory	Read data from a Radar Memory RAM has been corrupted	MTU	One of Multiple alarms will be generated based on MTU configuration
uncorrectable ECC error in a RAM internal to the BITMGR	Read data from an BITMGR RAM has been corrupted		

**Bit Manager (BITMGR)****Table 1105 SMU Events (cont'd)**

Event	Definition	Source	Comments
Register CRC check	Computed Register CRC does not match pre-programmed value	BITMGR	Alarm Mapping is defined in the SMU Specification
Radar Memory Control Logic Failure	The control signals for accessing Radar Memory are implemented as redundant logic. This alarm condition indicates that the BITMGR has detected an inconsistency in the redundant logic		
Radar Memory Access Address Error	An access to the Radar Memory has been mapped to a tile that has not been enabled for BITMGR accesses by writing to the EMEM tile control registers		
Radar Memory Read Data ECC Check	The read data for Radar Memory accesses is returned with an ECC computed from the access address and data. This is checked and an alarm set if the check fails		
Logic redundancy check failure	The internal core logic of BITMGR is implemented with full redundancy. This alarm indicates that the checker has detected a run time inconsistency between the two copies.		

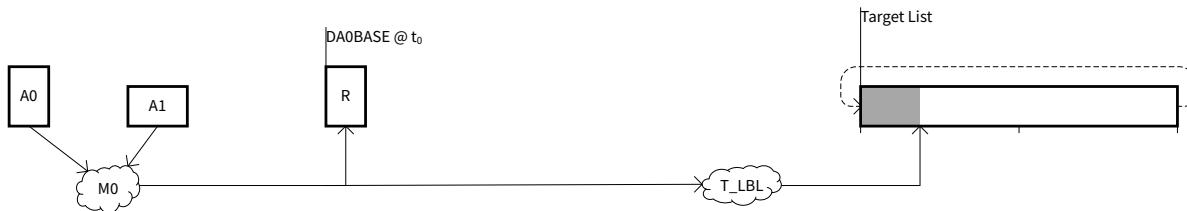
Note: *the SMU allows the user to define the action that will be taken as a result of these events (see SMU specification).*

## 21.8 Use Cases

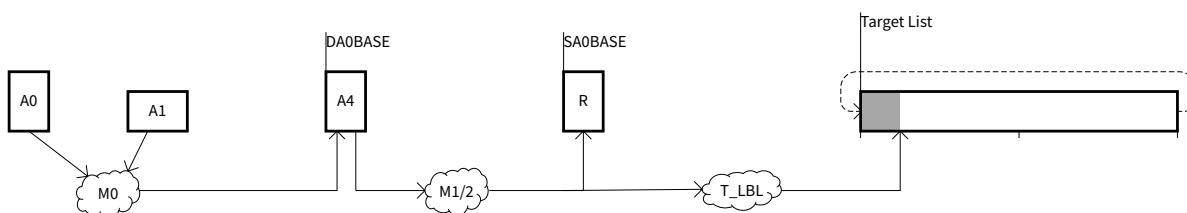
This section details assumed use case restrictions for the use of the BITMGR

## Bit Manager (BITMGR)

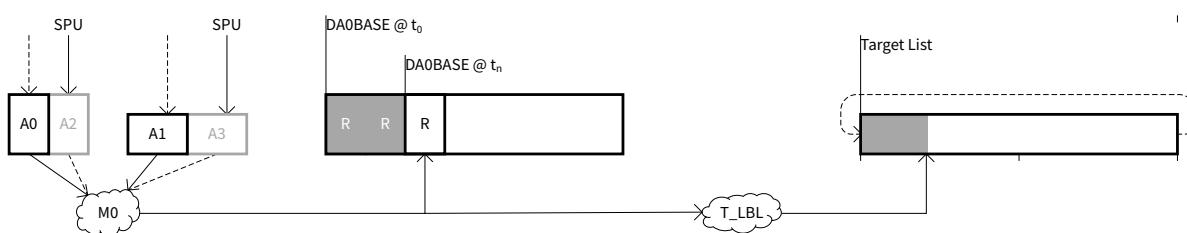
### UseCase – SW Single



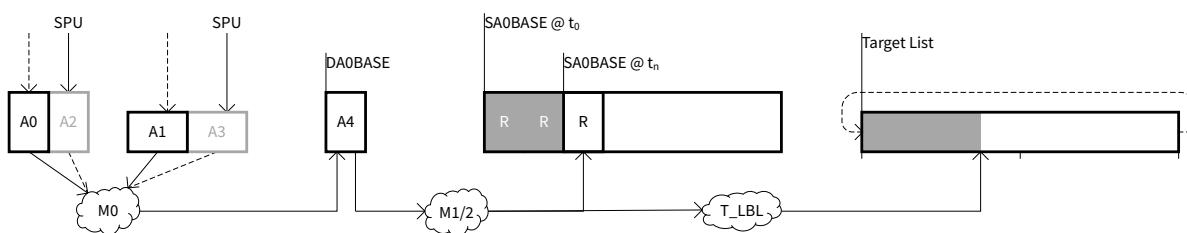
### UseCase – SW Single Chained



### UseCase – HW batch



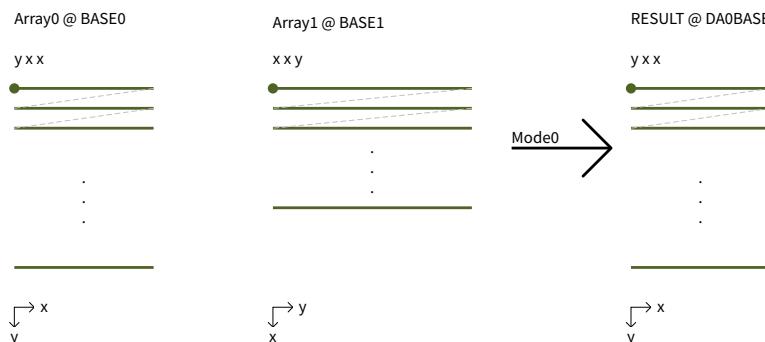
### UseCase – HW Chained batch



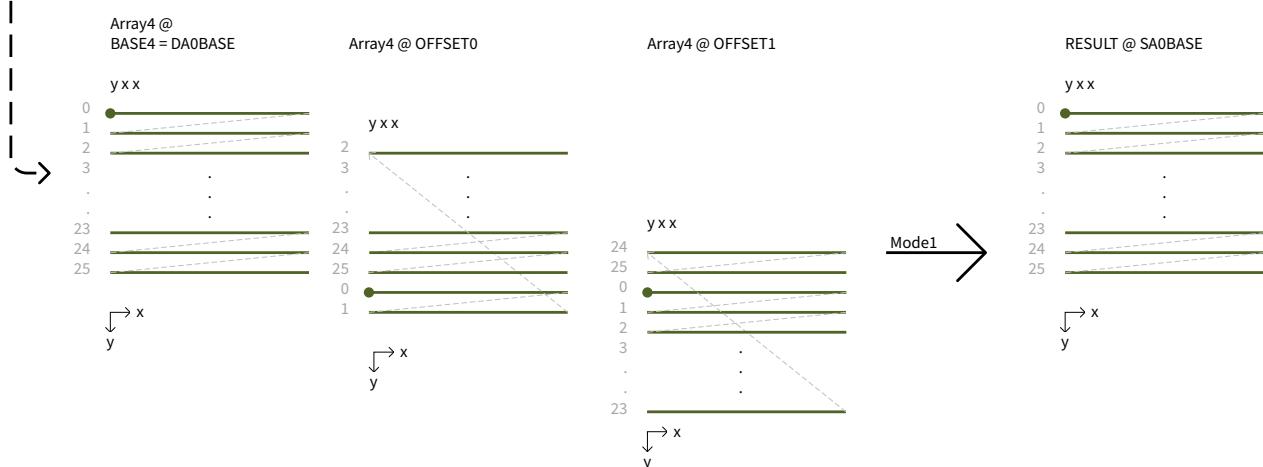
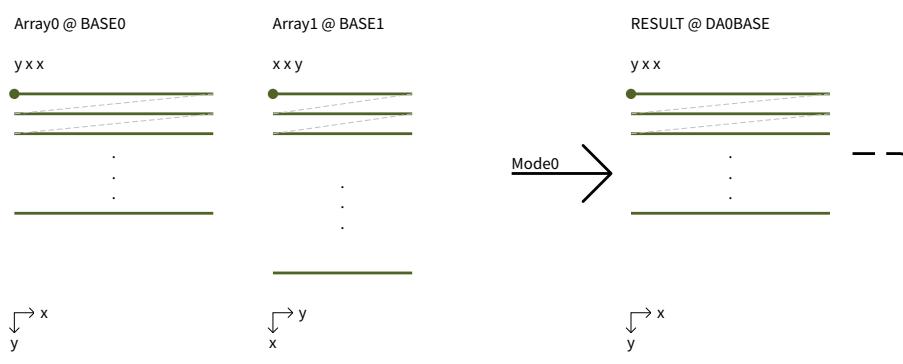
**Figure 276 Use case, process flow**

## Bit Manager (BITMGR)

### UseCase1



### UseCase2



**Figure 277 Use case assumptions**

**Bit Manager (BITMGR)****21.9 I/O Interfaces****Table 1106 List of BITMGR Interface Signals**

Interface Signals	I/O	Description
emem_rd		<b>BITMGR EMEM read interface</b> BITMGR EMEM read interface
emem_wr		<b>BITMGR EMEM Write Interface</b> BITMGR EMEM Write Interface
bitmgr_sfr		<b>BBB Slave Interface to the BITMGR Special Function Registers</b> BITMGR Module Configuration Registers
sx_irq_bitmgr		<b>BITMGR Interrupt Socket</b> BITMGR Interrupt Socket
sx_alarm_bitmgr		<b>BITMGR Alarm socket</b> BITMGR Alarm Socket
sx_ssh_com		<b>BITMGR Common SSH interface</b> BITMGR Common SSH interface
pbuf2a_0_ram		<b>BITMGR Private Buffer Ram SSH Interface</b> BITMGR Private Buffer Ram SSH Interface, First half of double buffer, a, bits 127:0
pbuf2a_1_ram		<b>BITMGR Private Buffer Ram SSH Interface</b> BITMGR Private Buffer Ram SSH Interface, First half of double buffer, a, bits 255:128
pbuf2b_0_ram		<b>BITMGR Private Buffer Ram SSH Interface</b> BITMGR Private Buffer Ram SSH Interface, Second half of double buffer, b, bits 127:0
pbuf2b_1_ram		<b>BITMGR Private Buffer Ram SSH Interface</b> BITMGR Private Buffer Ram SSH Interface, Second half of double buffer, b, bits 255:128
sx_ocds_periph_ctrl		<b>SCU OCDS Peripheral Control Interface</b> SCU OCDS Peripheral Control Interface
sx_ocds_otgb01		<b>BITMGR OTGB Interface</b> BITMGR OTGB Interface
sx_trace_sri		<b>SPU Trace Output</b> Trace Output for Data Processed by the BITMGR
sx_prot		<b>BITMGR protection Interface</b> BITMGR protection Interface

**Bit Manager (BITMGR)****21.10 Revision History****Table 1107 Document Revision History**

Reference	Change to Previous Version	Comment
<b>V1.0.5</b>		
<a href="#">Page 7</a>	Changes to the description based on the inclusion of new IS and SAF bits instead of implicit assumptions.	0000056734-155
<a href="#">Page 9</a>	Changes to the description base on the inclusion of the new IS and SAF bits instead of implicit assumptions.	0000056734-155
<a href="#">Page 10</a>	Added code fragments representing the packing of labels into list entry	0000056734-155
<a href="#">Page 13</a>	Details changes related to the independent operations of Mode 0 and Mode1/2.	0000056734-155
<a href="#">Page 13</a>	Added clarity to how the configuration affects the chained operation.	0000056734-155
<a href="#">Page 20</a>	Added a new base address register for Mode1/2.	0000056734-155
<a href="#">Page 23</a>	Defines a new bit IS	0000056734-155
<a href="#">Page 23</a>	Defines a new bit SAF	0000056734-155
<a href="#">Page 28</a>	Defines a new bit CHAIN	0000056734-155
<a href="#">Page 30</a>	A new register to separate out trigger and Status bits from the CTRL register.	0000056734-155
<a href="#">Page 36</a>	Reserved register space for Data CRC	0000056734-155
<a href="#">Page 36</a>	Reserved register space for Control CRC	0000056734-155
<a href="#">Page 37</a>	Reserved register for CRC of Control CRC registers	0000056734-155
<a href="#">Page 38</a>	Reserved register for Control CRC register masking.	0000056734-155
<a href="#">Page 54</a>	Reduced to reference already qualified modules.	0000056734-155
<a href="#">Page 52</a>	Debug chapter to augment register description.	0000056734-155
<a href="#">Page 53</a>	Description of Logical redundancy added.	0000056734-155
<a href="#">Page 55</a>	Description of Monitor CRC interface.	0000056734-155
<b>V1.0.6</b>		
-	No functional change	

## SPU Lockstep Comparator (SPULCKSTP)

### 22 SPU Lockstep Comparator (SPULCKSTP)

The SPU Lockstep Comparator module provides output comparators and difference detection for the two SPU instances. It is configured via the BBB.

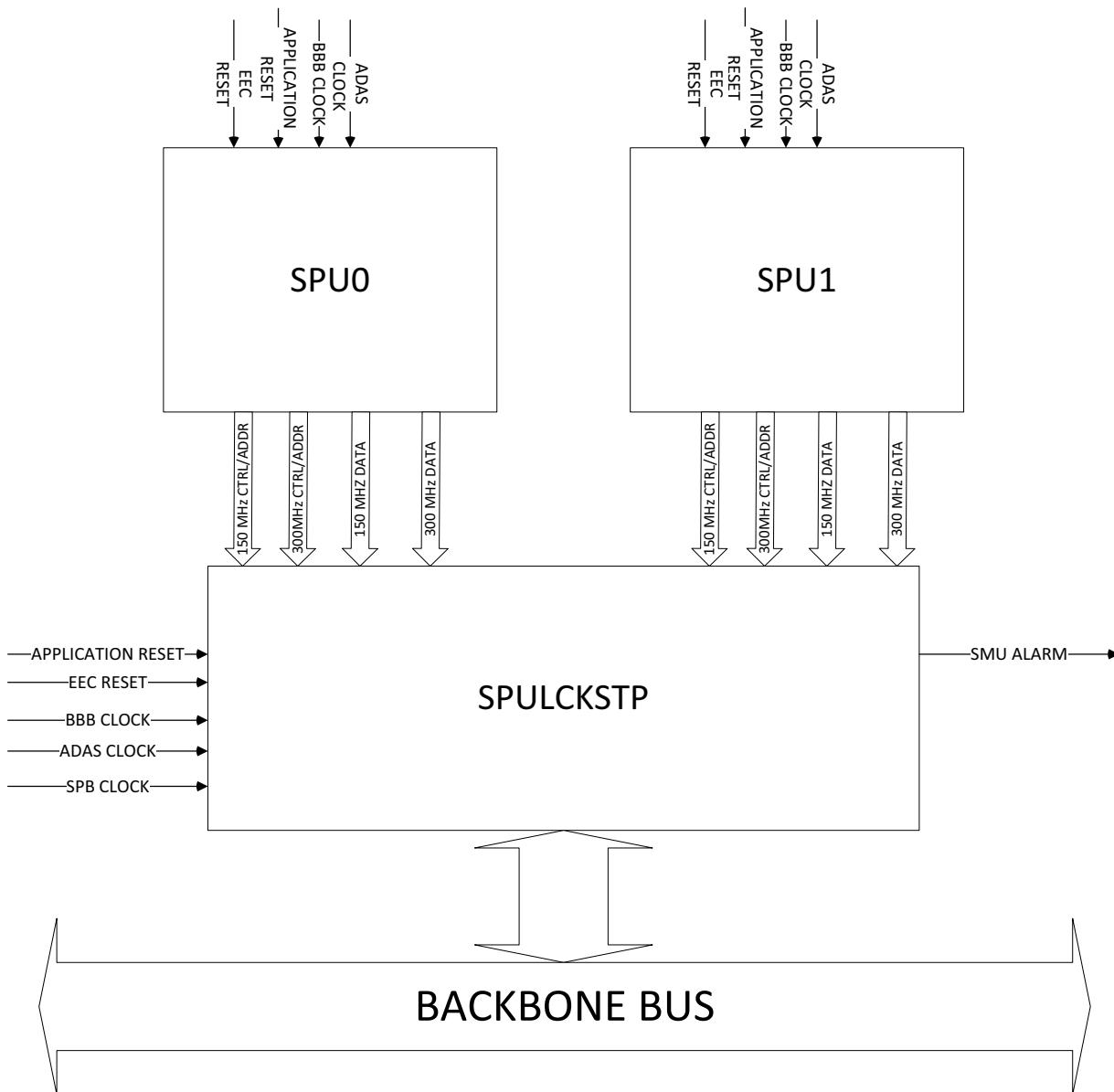


Figure 278 SPULCKSTP Interfaces

#### 22.1 Feature List

An overview of the features implemented in the SPU Lockstep Comparator follows:

- Control and Status Registers accessed via the BBB.
- Monitoring the output comparators for the two SPU instances and flagging any differences
- Running background, continuous self test on the lockstep comparators to validate the correct operation of the logic.

## SPU Lockstep Comparator (SPULCKSTP)

- Providing the necessary logic to trigger the two SPUs in the same clock cycle. This is necessary to allow the two SPUs to operate synchronously and allow the outputs to be successfully compared

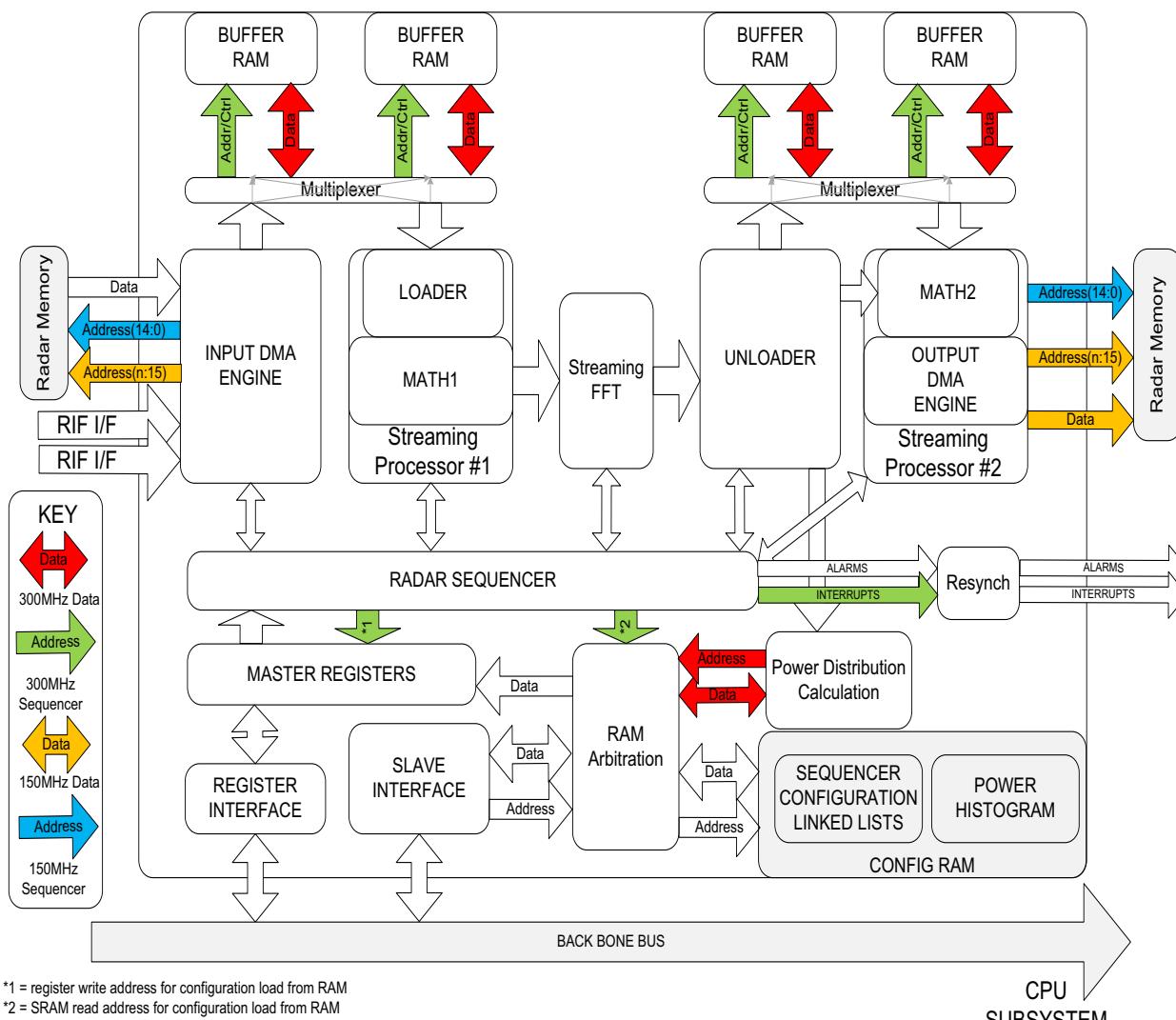
### 22.2 Overview

The SPU Lockstep implements four independent comparators. This allows for the two clock domains of the SPU, and also allows independent control of comparators on the address/control signals and the data signals. The SPU clock domains are:

- the 300MHz processing clock
- the 150 MHz BBB clock used for the interfaces to the RADAR memory (also known as EMEM),

The intention is to allow the operation sequence of identical SPU configurations to be checked even if the two SPU instances are processing different data. This is the “Partial Lockstep” mode. The normal operating mode where data signals are also checked is known as “Full Lockstep”.

**Figure 279** below shows the monitoring points associated with each of the four comparators.



**Figure 279 Lockstep Comparator Monitoring Points**

## SPU Lockstep Comparator (SPULCKSTP)

To enable partial lockstep, the 300MHz and 150MHz “sequencer” comparators are needed. To enable full lockstep, all four comparators are needed.

It is also a condition of use for the SPU that two SPU instances cannot use the same RADAR memory tile for data output. If this is attempted SPU1 write data will be lost. If the SPU instances are operating on different data then the read address used by the SPUs for retrieving data from RADAR memory will also be different. These operating constraints mean that only the offset addresses within the RADAR memory tiles will be compared for Partial Lockstep. This allows different tiles to be used for input data and output data for the two SPU instances. All address bits of the RADAR memory busses will be compared for full lockstep.

This split in the RADAR memory address bus is shown in [Figure 279](#). In the diagram “n” is used to indicate the MSB of the address bus. “n” depends on the amount of RADAR memory in the system (e.g. for 4 MiB of RADAR memory<sup>1)</sup>, “n” will be 17<sub>D</sub>).

Full lockstep assumes that SPU configurations, including RADAR memory addresses, are identical. In this case, the full addresses are compared. Since both SPUs will be reading and writing the same data, the accesses from the second SPU (SPU1) will be ignored.

## 22.3 Functional Description

### 22.3.1 SPU Lockstep Control

The lockstep functions are enabled by the LSENx bitfields in the control register ([CTRL](#)) of the SPULCKSTP module. Each lockstep comparator has its own instance of the LSEN bitfield.

These registers are initialised by an application reset. In the initialisation state, all lockstep comparators will be disabled. The lockstep function can be enabled by the application software writing a 10<sub>B</sub> to the LSENx bitfield.

Writes to the control register will be subject to the access enable protection mechanisms of the module.

Bitfields are allocated as follows:

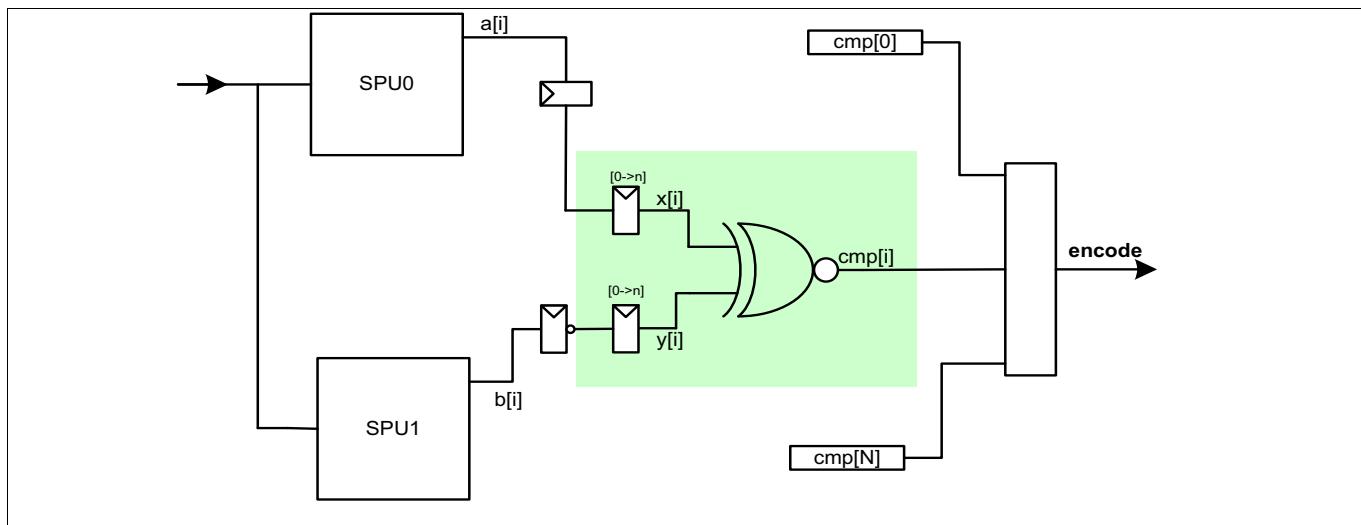
- CTRL.LSEN0: 300 MHz Sequencer Comparator
- CTRL.LSEN1: 150 MHz Sequence Comparator
- CTRL.LSEN2: 300 MHz Data Comparator
- CTRL.LSEN3: 150 MHz Data Comparator

### 22.3.2 SPU Lockstep Monitoring

The lockstep monitoring function will compare the outputs from the two SPU instances and report any difference that occurs to the Safety Management Unit (SMU) for appropriate action.

1) Radar Memory uses 256 bit words and the SPU uses word addressing

### SPU Lockstep Comparator (SPULCKSTP)



**Figure 280 Node Comparator**

**Figure 280** above shows the equivalent circuit of an arbitrary node comparator,  $i$ , of 0 to  $N$  comparators. The comparator monitors two signals, **a** and **b**, which are connected to the same output node of both SPU instances. The signals will be synchronised in the relevant instance using the instance clock to minimise impact on the instance timing.

After the optional synchronisation, one monitor point is inverted to reduce the risk of a common mode failure in the two monitored signals. The two inputs to the comparator are generated from **a** and **b**.

**x** which is equivalent to **a** delayed by  $n+1$  clock cycles, where  $n$  is an implementation dependent number of flip-flop stages inserted to allow timing closure

**y** which is equivalent to **b** delayed by  $n+1$  clock cycles.

As either **a** or **b** is inverted at the input to the delay stages, if the nodes differ (i.e. **x** and **y** are the same), the CMP signal is set to  $1_B$ .

The cmp[i] bus is scanned by two counters. One counter starts at the top index of the bus and the other at the bottom index. Each counter will stop when it encounters a bit of the cmp[i] bus which is indicating a failure. The counter values are then combined and output as the “encode” signal. This will cause the **encode** signal to flag the failing node and the failure will be detected. In the event that multiple nodes fail, the encode output will be the minimum and maximum values of all the indices,  $i$ , of the failing nodes. This has no impact on normal functioning but allows the self test logic to detect a real failure occurring in the same clock cycle that a fault is injected for test purposes.

The SPU Lockstep module transmits a single alarm to the SMU. The individual comparator which causes the alarm can be determined from the LCFAILx bitfields in the ERROR register. Bitfields are allocated as follows

- ERROR.LCFAIL0: 300 MHz Sequencer Comparator
- ERROR.LCFAIL1: 150 MHz Sequence Comparator
- ERROR.LCFAIL2: 300 MHz Data Comparator
- ERROR.LCFAIL3: 150 MHz Data Comparator

#### 22.3.3 Lockstep Self Test

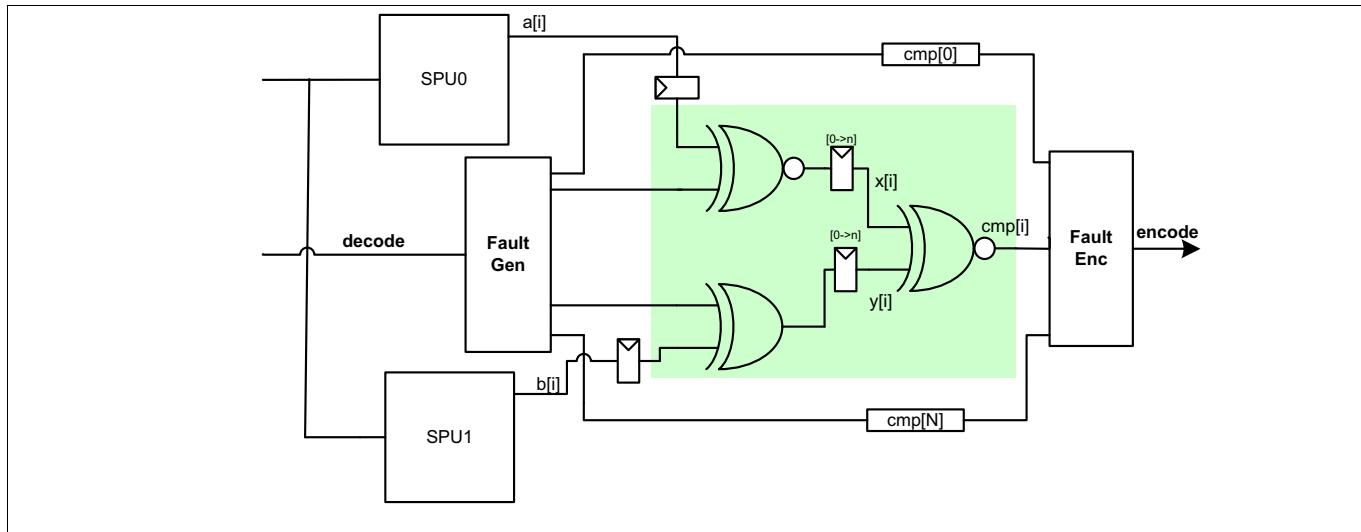
The SPU lockstep has a continuously running background self test of the lockstep comparator.

The self test function will inject faults into both inputs of each of the monitored nodes and verify that the fault is correctly detected by the monitoring logic.

## SPU Lockstep Comparator (SPULCKSTP)

In the event of a self test failure being detected, the failure will be reported to the Safety Management Unit for an appropriate response.

An equivalent circuit of a node comparator showing the fault injection logic is shown in [Figure 281](#)



**Figure 281 Node Comparator with Fault Injection**

Faults are injected using the **Fault Gen** block. This will use a binary number, **decode**, to calculate which node is to be tested. **decode** will be generated from a free running, binary counter using gray code number representation (the **input counter**). The Self test circuit will test every node at most once every 8192 clock cycles. Alternate test cycles will inject faults into either the **a** or **b** side of the comparator node. The complete self test cycle will therefore repeat at most every 16384 clock cycles.

Injecting a fault into either side of the comparator node will cause that node to fail unless a real fault occurs in the same clock cycle in which case the two faults will cancel out.

The node failing is processed by the **Fault Enc** block to generate a binary representation of the failing node, **encode**. This contains two numbers, the node index of the first node found to be failing counting up from the lowest minimum node index and the node index of the first node found to be failing counting down from the maximum node index.

If the lockstep block is functioning correctly, both values in **encode**, will either be 0 if no failures have been detected or the number of the node which has had a fault induced by the self test logic.

The values in **encode** are checked against a second, independent binary counter (the **monitor counter**). The **monitor counter** is also compared against the value of the **input counter**. In the event that either of the values in **encode** or the **input counter** fails to match the value of the monitor counter, a failure condition will be flagged to the SMU.

With this implementation, any of the following conditions will cause a self test fail:

- an actual fault occurring on any of the **a** or **b** nodes
- a stuck at  $0_B$  fault occurring on any of the **cmp** nodes
- a stuck at  $1_B$  fault occurring on any of the **cmp** nodes
- a failure in the **Fault Gen** block causing an incorrect or no fault to be injected
- a failure in the **Fault Enc** block causing an incorrect detection or no fault to be detected
- a stuck at fault on **x** (assuming that an injected fault does not always coincide with a masking pulse on **b**)
- a stuck at fault on **y** (assuming that an injected fault does not always coincide with a masking pulse on **a**)
- an actual fault on any of the **a** or **b** nodes coinciding with an injected fault

## SPU Lockstep Comparator (SPULCKSTP)

- a soft error occurring on either the **input counter** or **monitor counter**.

### 22.3.4 Functional Redundancy

All registers in the lockstep block which are not capable of being directly monitored for correct operation by the self test function will be duplicated. The duplicate register will store the inverse state of the master register. In the event of the duplicated registers not storing the inverse state value, an error will be flagged in the LCRFAILx fields of the ERROR register. The bitfields are allocated as follows

- ERROR.LCRFAIL0: 300 MHz Sequencer Comparator
- ERROR.LCRFAIL1: 150 MHz Sequence Comparator
- ERROR.LCRFAIL2: 300 MHz Data Comparator
- ERROR.LCRFAIL3: 150 MHz Data Comparator

The duplication extends to the register control bits (e.g. the LSEN bitfields).

### 22.3.5 Lockstep Failure Signalling Test

The lockstep comparator allows a failure to be injected into one of the comparator nodes to allow the signalling of failures to be verified. A failure can be injected by writing  $10_B$  to any of the LSTST bitfields of the TEST register. To avoid masking a real error, this test should only be conducted when the SPUs are idle.

- TEST.LSTST0: 300 MHz Sequencer Comparator
- TEST.LSTST1: 150 MHz Sequence Comparator
- TEST.LSTST2: 300 MHz Data Comparator
- TEST.LSTST3: 150 MHz Data Comparator

## SPU Lockstep Comparator (SPULCKSTP)

### 22.4 Registers

**Table 1108 Register Address Space - SPULCKSTP**

Module	Base Address	End Address	Note
SPULCKSTP	FA700000 <sub>H</sub>	FA7000FF <sub>H</sub>	SPU LOCKSTEP SFR Registers

**Table 1109 Register Overview - SPULCKSTP (ascending Offset Address)**

Short Name	Long Name	Offset Address	Page Number
CLC	Clock Control	000 <sub>H</sub>	7
MODID	Module Identification Register	004 <sub>H</sub>	8
CTRL	SPU Lockstep Control	010 <sub>H</sub>	9
ERROR	Error Monitoring Register	018 <sub>H</sub>	10
ERRCLR	Error Clear	01C <sub>H</sub>	10
TEST	Alarm Test Register	020 <sub>H</sub>	11
SPUCTRL	SPU Control	024 <sub>H</sub>	12
ACCENO	Access Enable Register 0	0E4 <sub>H</sub>	13
ACCEN1	Access Enable Register 1	0E8 <sub>H</sub>	14

#### 22.4.1 Details of SPULCKSTP Registers

##### Clock Control

<b>CLC</b> <b>Clock Control</b> <span style="float: right;">Reset Value: Table 1111</span>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>RES</b>															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>RES</b>												<b>EDIS</b>	<b>RES</b>	<b>DISS</b>	<b>DISR</b>
r												<b>rw</b>	<b>r</b>	<b>rh</b>	<b>rw</b>

Field	Bits	Type	Description
<b>DISR</b>	0	<b>rw</b>	<b>Disable Request</b> 0 <sub>B</sub> <b>Enable</b> , Request that the SPU Lockstep clock tree be switched on 1 <sub>B</sub> <b>Disable</b> , Request that the SPU Lockstep clock tree be switched off
<b>DISS</b>	1	<b>rh</b>	<b>Disable Status</b> This bit will be set to 1 if the SPU kernel clock is disabled 0 <sub>B</sub> <b>Enable</b> , The SPU Lockstep clock tree is switched on 1 <sub>B</sub> <b>Disabled</b> , The SPU Lockstep clock tree is switched off
<b>RES</b>	2, 31:4	<b>r</b>	<b>Reserved</b>

## SPU Lockstep Comparator (SPULCKSTP)

Field	Bits	Type	Description
<b>EDIS</b>	3	rw	<b>Sleep Mode Enable Control</b> Reserved. This bit currently has no effect on the SPU Lockstep. It should be kept at 0 for future compatibility 0 <sub>B</sub> <b>Disabled</b> , Sleep Mode is Off 1 <sub>B</sub> <b>Enable</b> , Sleep Mode is On (no effect)

**Table 1110 Access Mode Restrictions of CLC sorted by descending priority**

Mode Name	Access Mode		Description
ENDINIT and Master enabled in ACCEN	r	RES	Write Access During Initialisation
	rh	DISS	
	rw	DISR, EDIS	
Otherwise (default)	r	DISR, EDIS, RES	Default Read Access
	rh	DISS	

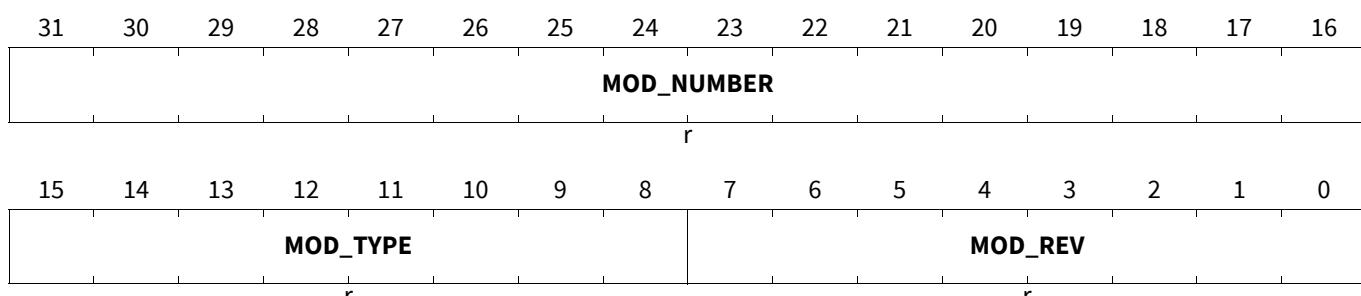
**Table 1111 Reset Values of CLC**

Reset Type	Reset Value	Note
Application Reset	0000 0003 <sub>H</sub>	Application Reset

### Module Identification Register

#### MODID

**Module Identification Register** (004<sub>H</sub>) "Application Reset" Value: 00E5 C001<sub>H</sub>



Field	Bits	Type	Description
<b>MOD_REV</b>	7:0	r	<b>Module Revision Number</b> This bit field defines the module revision number. The value of a module revision starts with 01H (first revision)
<b>MOD_TYPE</b>	15:8	r	<b>Module Type</b> Set to 0xC0 to indicate a 32 bit module
<b>MOD_NUMBE</b> R	31:16	r	<b>Module Number Value</b> Set to 0x00E5. This number is unique to the SPU Lockstep Module.

## SPU Lockstep Comparator (SPULCKSTP)

**Table 1112 Access Mode Restrictions of MODID sorted by descending priority**

Mode Name	Access Mode		Description
Write Accesses	r	MOD_NUMBER, MOD_REV, MOD_TYPE	Any Write Access will cause bus error. All reads permitted
Otherwise (default)	r	MOD_NUMBER, MOD_REV, MOD_TYPE	Default Access Mode. Read Only

### SPU Lockstep Control

Control Register for configuring the SPU Lockstep Safety Mechanism

#### CTRL

**SPU Lockstep Control** **Application Reset Value: 0005 0055<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
								r				rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES															
				r				rw		rw		rw		rw	

Field	Bits	Type	Description
LSENx (x=0-3)	2*x+1:2*x	rw	<b>Lockstep Comparator x Enable</b> Set this bitfield to 10 <sub>B</sub> to enable the comparator and 01 <sub>B</sub> to disable. Other values are invalid and will cause a status bitfield to be set (ERROR.LCRFAILx) 00 <sub>B</sub> <b>ERR0</b> , Invalid 01 <sub>B</sub> <b>DISABLE</b> , Disable Comparator 10 <sub>B</sub> <b>ENABLE</b> , Enable Comparator 11 <sub>B</sub> <b>ERR3</b> , Invalid
RES	15:8, 31:20	r	<b>Reserved</b>
ERRDIS	17:16	rw	<b>Error Disable</b> Disable Errors for SPU1 on arbitration Collision with SPU0 write access 00 <sub>B</sub> <b>ERR0</b> , Invalid 01 <sub>B</sub> <b>ENABLE</b> , Enable Arbitration Errors 10 <sub>B</sub> <b>DISABLE</b> , Disable Arbitration Errors 11 <sub>B</sub> <b>ERR3</b> , Invalid
MS	19:18	rw	<b>Mirror SPU0</b> The read data and acknowledge/error signals on the SPU1 EMEM read data interface can be mirrored from the SPU0 EMEM interface 00 <sub>B</sub> <b>ERR0</b> , Invalid 01 <sub>B</sub> <b>DISABLE</b> , Disable Interface Mirroring 10 <sub>B</sub> <b>ENABLE</b> , Enable Interface Mirroring 11 <sub>B</sub> <b>ERR3</b> , Invalid

## SPU Lockstep Comparator (SPULCKSTP)

**Table 1113 Access Mode Restrictions of CTRL sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rw	ERRDIS, LSENx (x=0-3), MS	
Otherwise (default)	r	ERRDIS, LSENx (x=0-3), MS, RES	Default Read Access

### Error Monitoring Register

#### ERROR

**Error Monitoring Register (018<sub>H</sub>) Application Reset Value: 0055 0055<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES								LCRFAIL3	LCRFAIL2	LCRFAIL1	LCRFAIL0				
				r				rh		rh		rh		rh	
RES								LCFAIL3	LCFAIL2	LCFAIL1	LCFAIL0				
				r				rh		rh		rh		rh	

Field	Bits	Type	Description
LCFAILx (x=0-3)	2*x+1:2*x	rh	<b>Lockstep Comparator x Fail</b> 00 <sub>B</sub> <b>ERR0</b> , Invalid State Error 01 <sub>B</sub> <b>OK</b> , No Failure Detected, Normal Operation 10 <sub>B</sub> <b>FAIL</b> , Failure Detected 11 <sub>B</sub> <b>ERR3</b> , Invalid State Error
RES	15:8, 31:24	r	<b>Reserved</b>
LCRFAILx (x=0-3)	2*x+17:2*x +16	rh	<b>Lockstep Comparator x Redundancy Fail</b> 00 <sub>B</sub> <b>ERR0</b> , Invalid State Error 01 <sub>B</sub> <b>OK</b> , No Failure Detected, Normal Operation 10 <sub>B</sub> <b>FAIL</b> , Failure Detected 11 <sub>B</sub> <b>ERR3</b> , Invalid State Error

**Table 1114 Access Mode Restrictions of ERROR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rh	LCFAILx (x=0-3), LCRFAILx (x=0-3)	
Otherwise (default)	r	RES	Default Read Access
	rh	LCFAILx (x=0-3), LCRFAILx (x=0-3)	

### Error Clear

This register allows the individual Error flags in the ERROR register to be cleared

## SPU Lockstep Comparator (SPULCKSTP)

**ERRCLR****Error Clear**(01C<sub>H</sub>)Application Reset Value: 00AA 00AA<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r								rwh		rwh		rwh		rwh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES															
r								rwh		rwh		rwh		rwh	

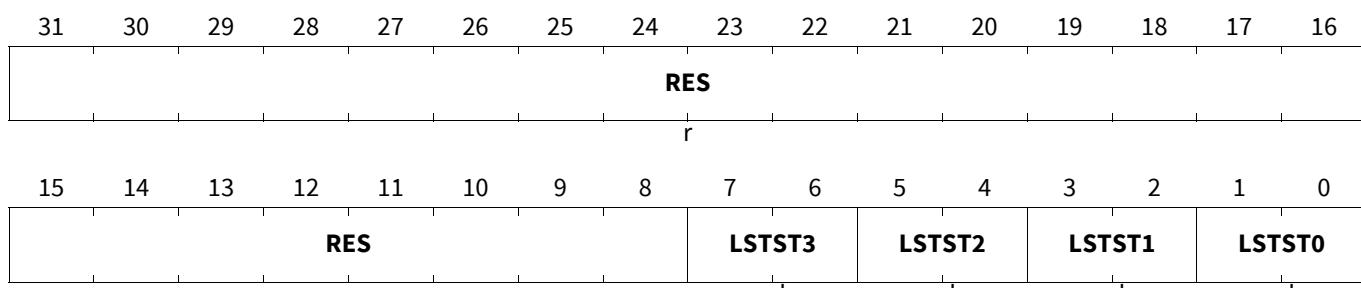
Field	Bits	Type	Description
CLR <sub>x</sub> (x=0-3)	2 <sup>x+1</sup> :2 <sup>x</sup>	rwh	<b>Clear Lockstep Comparator x Fail Flag</b> This field will always read as 10 <sub>B</sub> 00 <sub>B</sub> <b>ERR0</b> , Invalid 01 <sub>B</sub> <b>CLEAR</b> , Clear the Error Flag 10 <sub>B</sub> <b>NOP</b> , No Effect 11 <sub>B</sub> <b>ERR3</b> , Invalid
RES	15:8, 31:24	r	<b>Reserved</b>
RCLR <sub>x</sub> (x=0-3)	2 <sup>x+17</sup> :2 <sup>x</sup> +16	rwh	<b>Clear Lockstep Comparator x Redundancy Fail Flag</b> This field will always read as 10 <sub>B</sub> 00 <sub>B</sub> <b>ERR0</b> , Invalid 01 <sub>B</sub> <b>CLEAR</b> , Clear the Error Flag 10 <sub>B</sub> <b>NOP</b> , No Effect 11 <sub>B</sub> <b>ERR3</b> , Invalid

**Table 1115 Access Mode Restrictions of ERRCLR sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rwh	CLR <sub>x</sub> (x=0-3), RCLR <sub>x</sub> (x=0-3)	
Otherwise (default)	r	RES	Default Read Access
	rh	CLR <sub>x</sub> (x=0-3), RCLR <sub>x</sub> (x=0-3)	

**Alarm Test Register**

This register allows the individual comparators in the SPU Lockstep module to be tested

**SPU Lockstep Comparator (SPULCKSTP)****TEST****Alarm Test Register**(020<sub>H</sub>)Application Reset Value: 0000 0055<sub>H</sub>

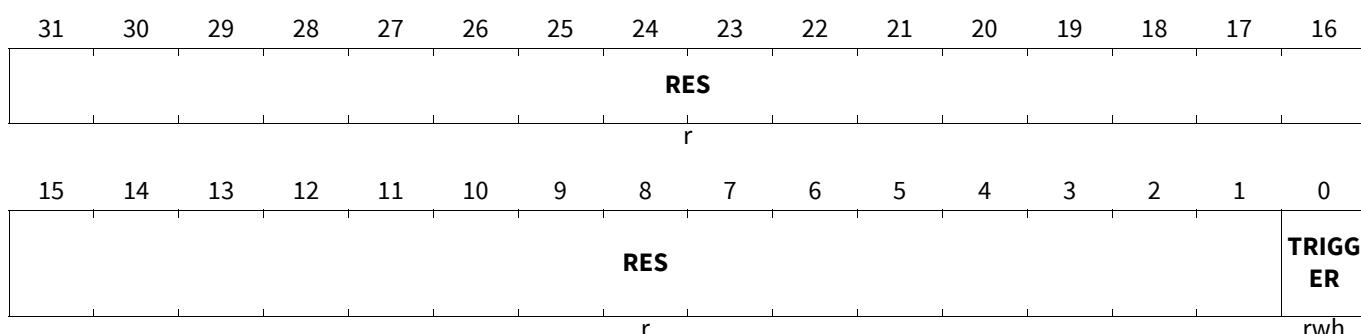
Field	Bits	Type	Description
LSTSTx (x=0-3)	2*x+1:2*x	rwh	<b>Test Lockstep Comparator x Alarm</b> This bitfield will always read as 01 <sub>B</sub> 00 <sub>B</sub> <b>ERR0</b> , Invalid 01 <sub>B</sub> <b>CLEAR</b> , No Effect 10 <sub>B</sub> <b>TEST</b> , Test the Alarm 11 <sub>B</sub> <b>ERR3</b> , Invalid
RES	31:8	r	<b>Reserved</b>

**Table 1116 Access Mode Restrictions of TEST sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rwh	LSTSTx (x=0-3)	
Otherwise (default)	r	RES	Default Read Access
	rh	LSTSTx (x=0-3)	

**SPU Control**

This register allows limited control of the SPU(s) in the product from the SPU\_LCKSTP module.

**SPUTCTRL****SPU Control**(024<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

## SPU Lockstep Comparator (SPULCKSTP)

Field	Bits	Type	Description
TRIGGER	0	rwh	<b>SPU Trigger</b> Write $1_B$ to this field to synchronously trigger both SPUs. The CTRL.MODE field in each SPU must be set to EXT for this to have any effect. This field will always read as $0_B$
RES	31:1	r	<b>Reserved</b> Reserved bits. Read as $0_B$ , should be written with $0_B$

**Table 1117 Access Mode Restrictions of [SPECTRL](#) sorted by descending priority**

Mode Name	Access Mode		Description
Master enabled in ACCEN	r	RES	Write Access for permitted masters only, writes for non-permitted masters cause bus error
	rwh	TRIGGER	
Otherwise (default)	r	RES	Default Read Access
	rh	TRIGGER	

### Access Enable Register 0

The Access Enable Register 0 controls access for transactions to registers with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The SPULCKSTP is prepared for an 6 bit TAG ID. The registers ACCENO / ACCEN1 are providing one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to ACCENO.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ... ,EN31 -> TAG ID 011111B.

#### ACCENO

#### Access Enable Register 0 (0E4H) Reset Value: [Table 1119](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

Field	Bits	Type	Description
ENx (x=0-31)	x	rw	<b>Access Enable for Master TAG ID x</b> This bit enables access to the SPU Lockstep Module register addresses for transactions with the Master TAG ID x $0_B$ <b>RO</b> , Write access will terminate without error but will not be executed. $1_B$ <b>RW</b> , Write and read accesses will be executed

## SPU Lockstep Comparator (SPULCKSTP)

**Table 1118 Access Mode Restrictions of ACCENO sorted by descending priority**

Mode Name	Access Mode		Description
Safety ENDINIT	rw	ENx (x=0-31)	Writes Enabled during Safety Initialisation
Otherwise (default)	r	ENx (x=0-31)	Default Access

**Table 1119 Reset Values of ACCENO**

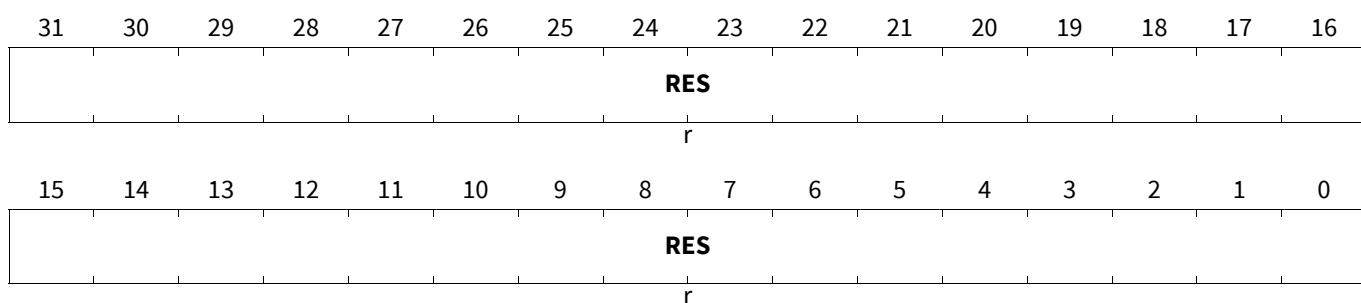
Reset Type	Reset Value	Note
Application Reset	FFFF FFFF <sub>H</sub>	Application Reset

### Access Enable Register 1

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). These tags are not used on the BBB and so no programmable bits are provided.

#### ACCEN1

**Access Enable Register 1** Reset Value: Table 1121



Field	Bits	Type	Description
RES	31:0	r	<b>Reserved</b> Read as 0, should be written with 0.

**Table 1120 Access Mode Restrictions of ACCEN1 sorted by descending priority**

Mode Name	Access Mode		Description
Safety ENDINIT	r	RES	Writes Enabled during Safety Initialisation
Otherwise (default)	r	RES	Read Access Only. Writes will be silently ignored

**Table 1121 Reset Values of ACCEN1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	(application_reset)

## 22.5 Use Cases

This section describes the use of the SPU Lockstep Module.

## SPU Lockstep Comparator (SPULCKSTP)

### 22.5.1 Conditions of Use

The main assumption of use is that the SPU module will be in a known and deterministic state when it is idle. Its state changes, and therefore the sequence of output states feeding onto the lockstep comparators, after leaving the idle state will be determined entirely by the configured register settings. There are two exceptions to this where the data values can affect the flow of operation:

- Bin Rejection based on either CFAR or Local MAX
- Power Histogram

The Power Histogram signals are excluded from the “Partial Lockstep” comparison for this reason. Bin Rejection affects the output interface to the RADAR memory and this interface is not excluded from “Partial Lockstep”. Bin Rejection cannot therefore be used with “Partial Lockstep”.

### 22.5.2 Set Up

The write interfaces to both the registers and the configuration memory from the bus are outside the zone of duplication and can therefore be configured as normal. Both SPU instances need to be configured identically. The lockstep comparators should only be enabled when the SPUs are idle, so this should be done once configuration is complete but before the SPUs are triggered.

The comparators can be disabled at any time.

#### 22.5.2.1 Specific Setup for Full Lockstep

The use of full lockstep requires both SPUs to be reading the same input data and writing to the same addresses in RADAR memory in exactly the same clock cycles. For this to work, two configuration changes need to be made to the RADAR memory configuration

- Write accesses from SPU1 to the RADAR memory must fail silently.
  - Errors on an arbitration collision can be disabled for SPU1 by changing the CTRL.ERRDIS bitfield from its default value of  $01_B$  to  $10_B$ .
- Read accesses from SPU1 to RADAR memory must be ignored. Data returned to SPU0 must be echoed onto the SPU1 pins.
  - This operating mode is enabled by changing the CTRL.MS bitfield from its default value of  $01_B$  to  $10_B$ .

### 22.5.3 SPU Triggering

The SPUs must start processing in the same clock cycle for the lockstep to work as intended. This is done by setting the SPUs to external trigger (SPU\_CTRL.MODE=EXT) and starting processing by writing a  $1_B$  to the TRIG bitfield of the SPUTRIG register.

When the data is being sourced from both RIFs, the RIFs also need to be configured to operate in a synchronous manner by setting RIF\_RSM0.LCKSTP. This ensures that both RIFs consistently output data on the same clock edge.

### 22.5.4 Expected Use Cases

Three specific use cases have been anticipated when developing this module. Other use cases are possible but behaviour may not be as expected.

## SPU Lockstep Comparator (SPULCKSTP)

**Table 1122 SPU Lockstep Use Cases**

<b>Configuration</b>	<b>CTRL</b>					
	<b>LSEN0</b>	<b>LSEN1</b>	<b>LSEN2</b>	<b>LSEN3</b>	<b>MS</b>	<b>ERRDIS</b>
Off	01 <sub>B</sub>					
Partial	10 <sub>B</sub>	10 <sub>B</sub>	01 <sub>B</sub>	01 <sub>B</sub>	01 <sub>B</sub>	01 <sub>B</sub>
Full	10 <sub>B</sub>					

---

**SPU Lockstep Comparator (SPULCKSTP)****22.6 Revision History****Table 1123 Revision History**

Reference	Change to Previous Version	Comment
<b>V1.2.5</b>		
-	Files regenerated for new project baseline.	

## Extension Memory (EMEM)

### 23 [[[Extension Memory (EMEM)]]]

[The EMEM (at [Figure 282]) contains RAM blocks (EMEM Tiles) which can be alternatively used for application (ADAS), calibration or trace data storage. The EMEM has interfaces to MCDS, SEP, SRI and BBB. ]

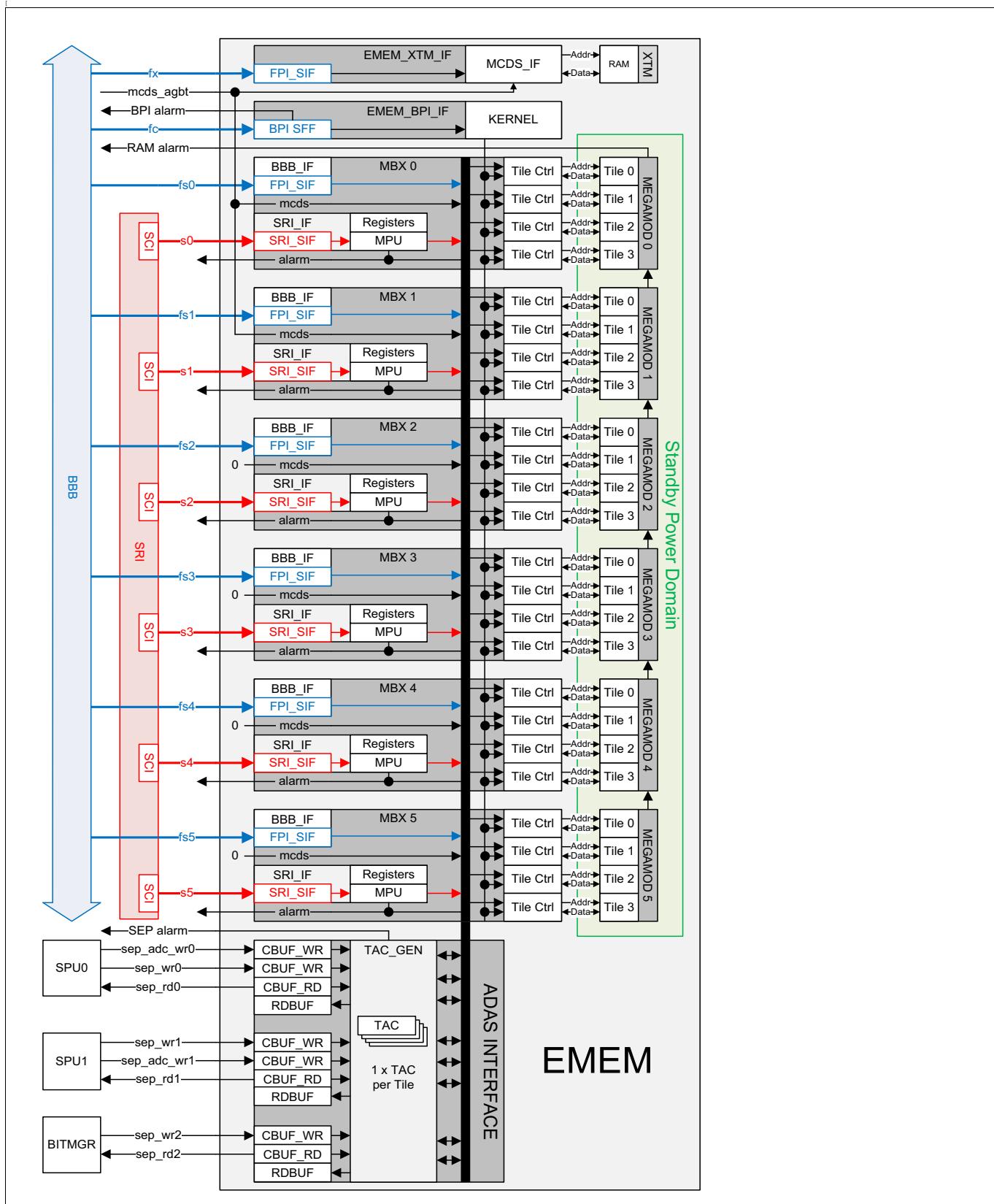


Figure 282 [Block Diagram of EMEM]]

**Extension Memory (EMEM)****[[EMEM Glossary]]****Table 1124 [[EMEM Acronyms[ ]]]**

[Acronym]	[Description]
[AGBT]	[Aurora Gigabit Trace Interface]
[EMEM]	[Extension Memory]
[MCDS]	[Multi Core Debug Solution]
[MPU]	[Memory Protection Unit]
[MTU]	[Memory Test Unit]
[NTN]	[Next Tile Now]
[SECDED]	[Single Error Correction Double Error Detection]
[SEP]	[SPU to EMEM Protocol]
[TCM]	[Trace or Common Memory]
[XCM]	[Extended Common Memory]
[XTM]	[eXtra Trace Memory]

**Table 1125 [[EMEM Terms[ ]]]**

[Acronym]	[Description]
[EMEM Core]	[An EMEM core contains the EMEM configuration interface and registers.]
[EMEM Module]	[An EMEM module consists of four EMEM tiles. Potentially each EMEM module has four interfaces propagated down to each EMEM tile: <ul style="list-style-type: none"><li>• [[MCDS interface (TCM and XTM)]]</li><li>• [SEP interface (TCM and XCM)]</li><li>• [SRI slave interface (TCM and XCM)]</li><li>• [BBB slave interface (TCM, XTM and XCM)]]</li></ul>
[EMEM Tile]	[An EMEM tile consists of 256 Kbyte RAM with SECDED capability.]
[XTM Tile]	[An XTM tile consists of 8 Kbyte RAM with SECDED capability.]

**23.1 [[Feature List]]**

[The EMEM has the following features:]

- [[Software may configure the operation mode of each individual **[EMEM Tile]**.]]
- [16 Kbyte XTM for trace data only.]
- [Standby power supply for TCM and XCM.]]

[The EMEM supports the following applications:]

- [[Application/ADAS data]]
- [Program code]
- [Calibration data]
- [Measurement data]
- [Trace data]
- [ED Prolog Code]]]

## Extension Memory (EMEM)

### 23.2 [[Overview]]

[Two different types of [EMEM Module] are available: ]

- [[[TCM]] may be split into Application/ADAS, Calibration and Trace Memory parts.]
- [[XCM]] may be used as Application/ADAS or Calibration Memory only.]

[The [XTM] is used for tracing without reducing the Common Memory. The XTM has two 8 Kbyte Tiles, which is sufficient for continuous trace over DAP.]

[Devices with AGBT use XTM0 and XTM 1 as an AGBT Trace buffer.]

### 23.3 [[Functional Description]]

[The EMEM configurations (at [Table 1126]) are instanced on AURIX TC3xx devices.]

**Table 1126 [EMEM Configurations]**

[Device]	[TC39x]	[TC37xEXT]	[TC35x]	[TC33xEXT]	[TC3Ax]
[Isolation Logic]	[Yes]	[Yes]	[No]	[No]	[No]
[TCM Size]	[2048 Kbyte]	[2048 KByte]	[2048 Kbyte]	[1024 Kbyte]	[2048 Kbyte]
[XCM Size]	[2048 Kbyte]	[1024 KByte]	[ - ]	[ - ]	[4096 Kbyte]
[XTM Size]	[16 Kbyte]				
[Total RAM Size]	[4096 + 16 Kbyte]	[3072 + 16 Kbyte]	[2048 + 16 Kbyte]	[1024 + 16 Kbyte]	[6144 + 16 Kbyte]
[SPU0 sep_adc_wr0]	[Yes]	[No]	[Yes]	[Yes]	[Yes]
[SPU0 sep_wr0]	[Yes]	[No]	[Yes]	[Yes]	[Yes]
[SPU0 sep_rd0]	[Yes]	[No]	[Yes]	[Yes]	[Yes]
[SPU1 sep_adc_wr1]	[Yes]	[No]	[Yes]	[No]	[Yes]
[SPU1 sep_wr1]	[Yes]	[No]	[Yes]	[No]	[Yes]
[SPU1 sep_rd1]	[Yes]	[No]	[Yes]	[No]	[Yes]
[BITMGR sep_wr2]	[No]	[No]	[No]	[No]	[Yes]
[BITMGR sep_wr2]	[No]	[No]	[No]	[No]	[Yes]

#### 23.3.1 [[Isolation Logic]]

[If the EMEM configuration (at [Table 1126]) instances [Isolation Logic], the [Isolation Logic] is implemented around each [EMEM Module] to prevent any power drain via powered gate outputs into the unpowered section of the device.]

[The logic isolating the inputs into the standby domain are controlled by the power on reset ([PORST]).]

[The logic isolating the outputs from the standby domain are controlled by the standby locked mode signal.]

Note: [[The local input to the EMEM to control the inputs to the standby domain is enable\_core\_ed\_i. This is driven from the EVR power on reset circuitry via a level shifter]]]

#### 23.3.2 [[EMEM Modes]]

[The EMEM shall be in one of the following modes:]

- [[[Standby Locked Mode].]
- [[Locked Mode].]

## Extension Memory (EMEM)

- [Unlocked Mode.]

[The mode of the EMEM depends on the power supply and the status flag SBRCTR.STBLOCK:]

**Table 1127 [EMEM Modes depending on V[ ]<sub>DD</sub> , V[ ]<sub>DDSB</sub><sup>1)</sup> and SBRCTR.STBLOCK[ ]]**

[SBRCTR.STBLOCK]	[V/ <sub>DDSB</sub> Only]	[V[ ] <sub>DD</sub> [ Only]]	[V[ ] <sub>DD</sub> [ and V[ ] <sub>DDSB</sub> ]]
[Locked Mode]	[Standby Locked Mode]	[Allowed]	[Locked Mode]
[Unlocked Mode]	[Not allowed]	[Not allowed]	[Unlocked Mode]

1) EMEM Standby Power Domain

### [[RAM Isolation]]

[The RAM shall be isolated prior to removing the power supplies. This assures that spikes or uncontrolled signal patterns applied to its interfaces during power cycling of the logic will not corrupt the content.]

[It is not allowed to remove the EMEM isolation logic when V[ ]<sub>DDSB</sub> is not powered since this will damage the chip.]

[Software and tools may monitor SBRCTR.STBPON to determine the availability of V[ ]<sub>DDSB</sub>.]

[If V[ ]<sub>DD</sub> is powered and V[ ]<sub>DDSB</sub> is not powered, the EMEM may report ECC errors.]

### 23.3.2.1 [[Locked Mode]]

[After EEC reset, the EMEM is in **Locked Mode**. If the EMEM is in **Locked Mode** then accesses from chip interfaces (MCDS, SEP, SRI and BBB) are treated as errors (at **Chapter 23.3.3.4**). Software shall configure the SBRCTR register to change the EMEM Mode. Both Unlocked Mode and the **Standby Locked Mode** may only be entered from Locked Mode.]

### 23.3.2.2 [[Standby Locked Mode]]

[As long as the standby power supply (V[ ]<sub>DDSB</sub>) is powered, the content of the RAM shall be retained.]

[On chip bus accesses to the **EMEM Core** CLC, ID and SBRCTR registers and **EMEM Module** registers are serviced normally. All other register and RAM accesses are terminated with error.]

### 23.3.2.3 [[Changing the EMEM Mode]]

[The mode of the EMEM shall be changed as follows:]

#### [[Transition from Locked Mode to Unlocked Mode]]

[To enter **Unlocked Mode**, software shall write the following sequence to the SBRCTR register:]

- [[[0000 0002<sub>H</sub>],]]
- [[0000 0006<sub>H</sub>],]
- [[0000 000E<sub>H</sub>] ]]

[If software writes any other value to SBRCTR before the complete sequence is written then the state machine shall be reset. To enter **Unlocked Mode**, the complete sequence shall be started again. If software writes to another register, the state machine sequence checking shall not be influenced. The current state is signalled with bit SBRCTR.STBLOCK.]

#### [[Transition from Unlocked Mode to Locked Mode]]

[To enter **Locked Mode** from **Unlocked Mode**, software shall write [0000 0090<sub>H</sub>] to the SBRCTR register.]

Note: [[The EMEM may only enter Locked Mode when there are no ongoing or pending accesses.]]]

## Extension Memory (EMEM)

### [[Transition from Locked Mode to Standby Locked Mode]]

[If the EMEM is in [Locked Mode] then [Standby Locked Mode] shall be entered by setting the [PORST] pin low and removing all supplies apart from VDDSB. Setting [PORST] low shall maintain the isolation of the standby domain after the supplies are removed.]]

### [[Transition from Standby Locked Mode to Unlocked Mode]]

[The transition from [Standby Locked Mode] to [Unlocked Mode] shall be performed with the following steps:]

1. [[The [PORST] pin shall be active.]
2. [All the power supplies shall be stable.]
3. [After the [PORST] release the EMEM shall be in [Locked Mode].]
4. [Software shall check the standby power domain is available (SBRCTR.STBPON = [1<sub>B</sub>]).]
5. [Software may configure [Unlocked Mode] (see [Changing the EMEM Mode], Transition from Locked Mode to Unlocked Mode). ]
6. [The [Isolation Logic] around the standby domain shall be switched off.]]]]]

### 23.3.3 [[Tile Modes]]

[After EEC reset, all EMEM Tiles shall be in [Unused Mode] (at [Figure 283]). Software shall configure the TILECONFIG register to assign [TCM] tiles to Common Memory, Trace Memory or Unused Mode and [XCM] tiles to Common Memory or Unused Mode. An [EMEM Tile] assigned to Trace Memory shall not be re-configured for Common Memory directly, or vice versa, but only after having been intermediately set to [Unused Mode].]

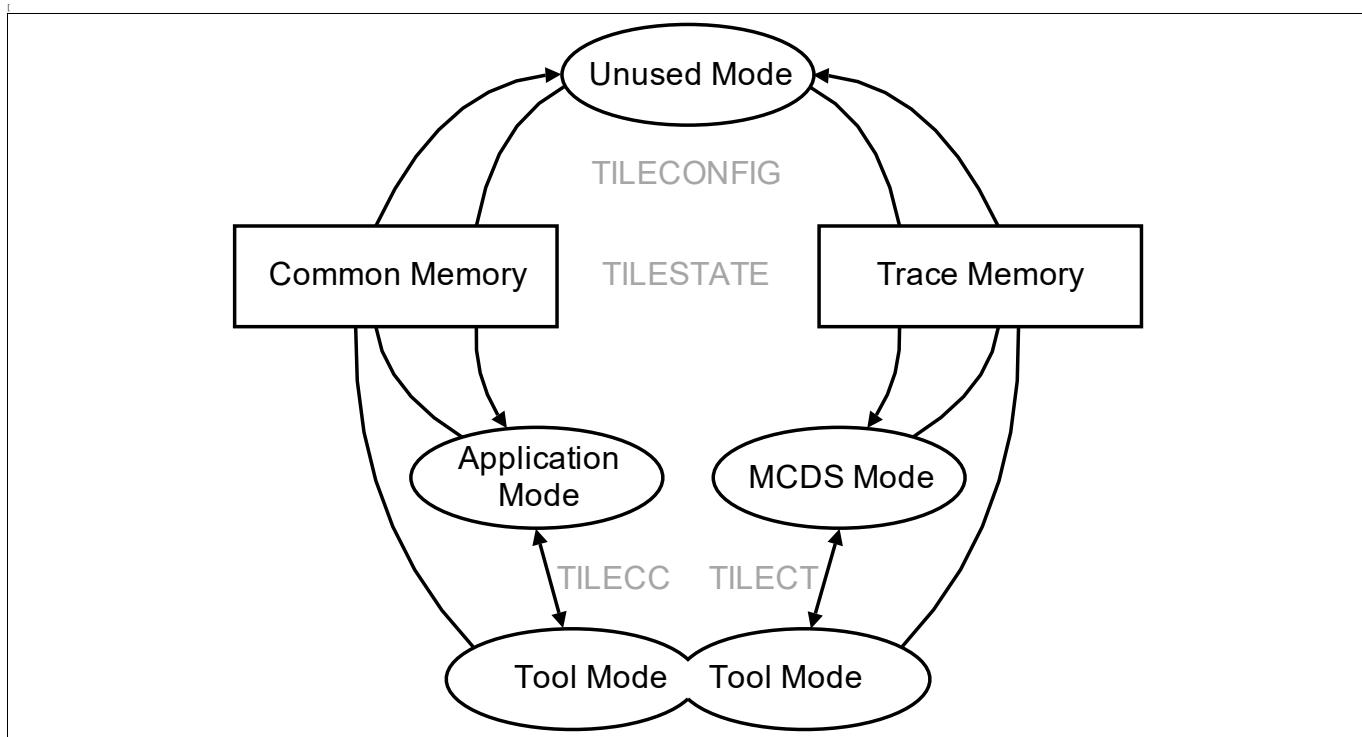
[An [EMEM Tile] assigned to Common Memory shall initially be in [Application Mode], and an [EMEM Tile] assigned to Trace Memory shall initially be in [MCDS Mode]. The assignment of each Tile is indicated by the TILESTATE register.]

[Each [EMEM Tile] assigned to Common Memory shall be individually configured between [Application Mode] and [Tool Mode]. The status of Tiles assigned to Common Memory is indicated by the TILECC register.]

[Each [EMEM Tile] assigned to Trace Memory shall be individually configured between [MCDS Mode] and [Tool Mode]. The status of Tiles assigned to Trace Memory is indicated by the TILECT register.]

[If an [EMEM Tile] is in [Unused Mode], then the settings in the TILECC and TILECT registers have no effect.]

## **Extension Memory (EMEM)**



**Figure 283 [Tile States and Modes]]**

### **23.3.3.1 [[Application Mode]]**

[If an [EMEM Tile] is assigned to Application Mode then read and write accesses may be performed by the SEP and SRI interfaces in an arbitrary sequence. The internal EMEM arbitration assigns higher priority to the SEP interface. When configuring writes to EMEM, each SEP write source should be allocated to a different [EMEM Tile]. ]

[For multiple simultaneous SEP accesses to an [EMEM Tile](#) see [TC39xED SEP Accesses to EMEM Tiles](#) for behavior.]

**Note:** [[The application should only assign EMEM tiles to Common Memory which are actually used. This protects the EMEM tile from unintended tool accesses via DAP/JTAG and BBB. It also allows an automatic cooperative sharing of the EMEM between an application and tool(s).]]

### 23.3.3.2 [[MCDS Mode]]

[The MCDS shall store trace messages in Trace Memory. The MCDS requires a consecutive selection of EMEM tiles as a trace buffer. The configuration of EMEM shall be consistent with the MCDS configuration concerning the address ranges. An [EMEM Tile] in MCDS Mode can only be written from the MCDS with trace data.]

**Note:** [For tracing the OSCU control bit OSTATE.EECDIS must be inactive (refer to OCDS documentation).]]

### 23.3.3.3 [[Tool Mode]]

[A tool shall check the configuration of the EMEM tiles to verify there are no conflicts in the EMEM configuration. A tool shall check for EMEM tiles in Unused Mode as indicated by the EMEM TILESTATE register and shall configure these EMEM tiles as part of the Common Memory in Tool Mode.]

**Note:** *[[It is important that a DAP/JTAG tool shall always access the EMEM via the IOC32P/E and not via Cerberus. This ensures that the tool cannot unintentionally corrupt a Tile in Application Mode.]]*

## Extension Memory (EMEM)

### [[Independent Calibration and Debug Tool Operation]]

[The two step approach to globally assigning EMEM tiles to Common Memory ([**Application Mode**]/Tool Mode) or Trace Memory ([**MCDS Mode**]/Tool Mode) supports safe operation with independent calibration and debug tools. The following rules apply:]

- [[A calibration tool shall only change the tile state between Common Memory and [**Unused Mode**.]]
- [A trace tool shall only change the tile state between Trace Memory and [**Unused Mode**.]]]

[The only resource shared by the calibration and debug tools is the EMEM TILECONFIG register. The tool(s) shall exercise care when changing the configuration of the [**MCDS**] and [**XCM**] tiles.]]

### 23.3.3.4 [[Accessing Tiles in Different Modes]]

[As soon as an EMEM interface receives an access request to an [**EMEM Tile**], the EMEM shall determine if the requesting interface has access rights. The EMEM shall check the mode of the addressed [**EMEM Tile**].]

■

**Table 1128 [Tile Access Options and Error Signaling[ ]]**

[Mode]	[SEP Interface]	[MCDS Interface]	[SRI Interface]	[BBB Interface]
[Application Mode]	[Access <sup>1)</sup> ]	[Error signalled]	[Access <sup>1)</sup> ]]	[BBB error]
[MCDS Mode]	[Error signalled]	[Access <sup>1)</sup> ]]	[SRI error]	[BBB error]
[Tool Mode]	[Error signalled]	[Error signalled]	[Access <sup>1)</sup> ]]	[Access <sup>1)</sup> ]]
[Unused Mode]	[Error signalled]	[Error signalled]	[SRI error]	[BBB error]

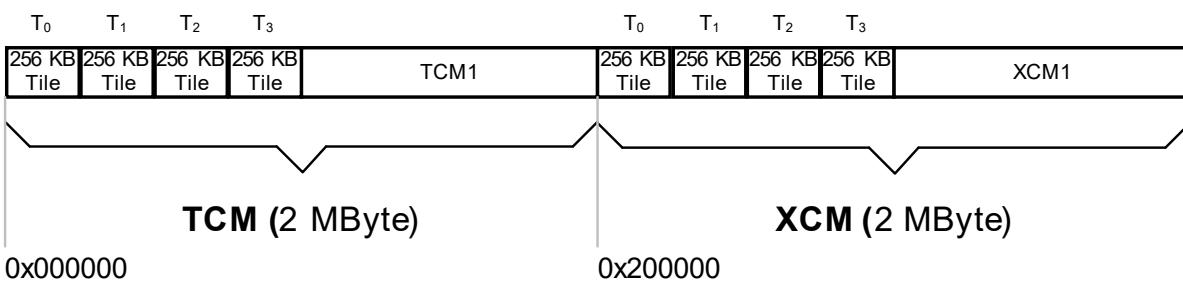
1) If an ECC error is detected then an access error is signalled to the corresponding interface.

### 23.3.4 [[Address Map]]

[The EMEM address map shall be such that any 1024 Kbyte TCM modules are contiguous. All 1024 Kbyte modules shall start at a 1024 Kbyte address offset. The 16 Kbyte XTM organized as 2 x 8 Kbyte tiles is located at the next 1024 Kbyte address boundary after the TCM and XCM modules.]

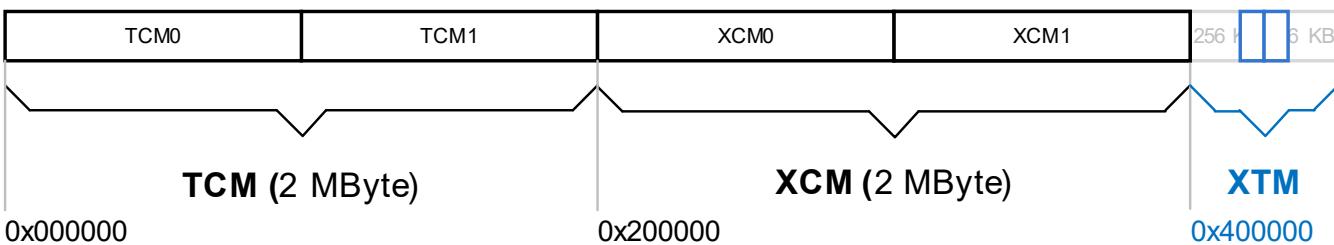
## Extension Memory (EMEM)

### Offset Address View from SEP and SRI



### Offset Address View from BBB

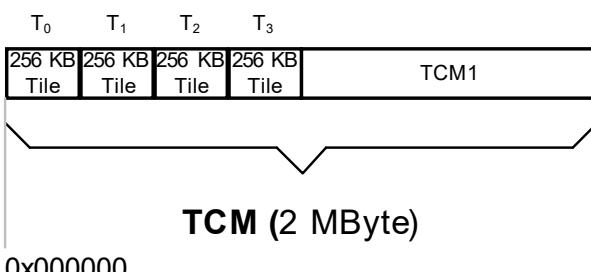
8 Kbyte XTM Tile mirrored in 256 Kbyte region



### Offset Address View from MCDS

XTM0 (8 KB address mirroring); AGBT Trace Tile  
XTM1 (address mirroring)

XTMx overlaid if XTMx is in Trace Mode .



### MCDS Overlay

If TILECT.XTM0 = 0  
XTM0 is overlaid to TCM Tile 0

If TILECT.XTM1 = 0  
XTM1 is overlaid to TCM Tile 1

Figure 284 [Offset Address View of TCM, XCM and XTM]]

#### 23.3.4.1 [[Address View]]

[The assignment of EMEM tiles to Trace Memory should start with Tile index 0 to maintain a continuous address range. The address view shows an example EMEM configuration and the address view from each interface.]]

#### 23.3.4.2 [[XTM Addressing]]

[The XTM shall be used in one of two modes indicated by the EMEM TILECT register:]

## Extension Memory (EMEM)

### 23.3.4.2.1 [[MCDS Mode]]

[The MCDS interface shall see an 8 Kbyte physical XTM tile address mirrored into the 256 Kbyte address space of a TCM tile. The MCDS may access the XTM. The DAP shall utilize the MCDS [NTN] capability when the XTM is used to generate a continuous MCDS trace output. NTN allows the MCDS trace output to be switched from one XTM tile to the other XTM tile before the complete 8 Kbyte XTM tile is written.]

#### [[AGBT ]]

[If AGBT is activated, XTM Tile0 and XTM Tile1 are used as an AGBT FIFO buffer.]

[The MCDS interface shall be able to write the EMEM Tiles.]

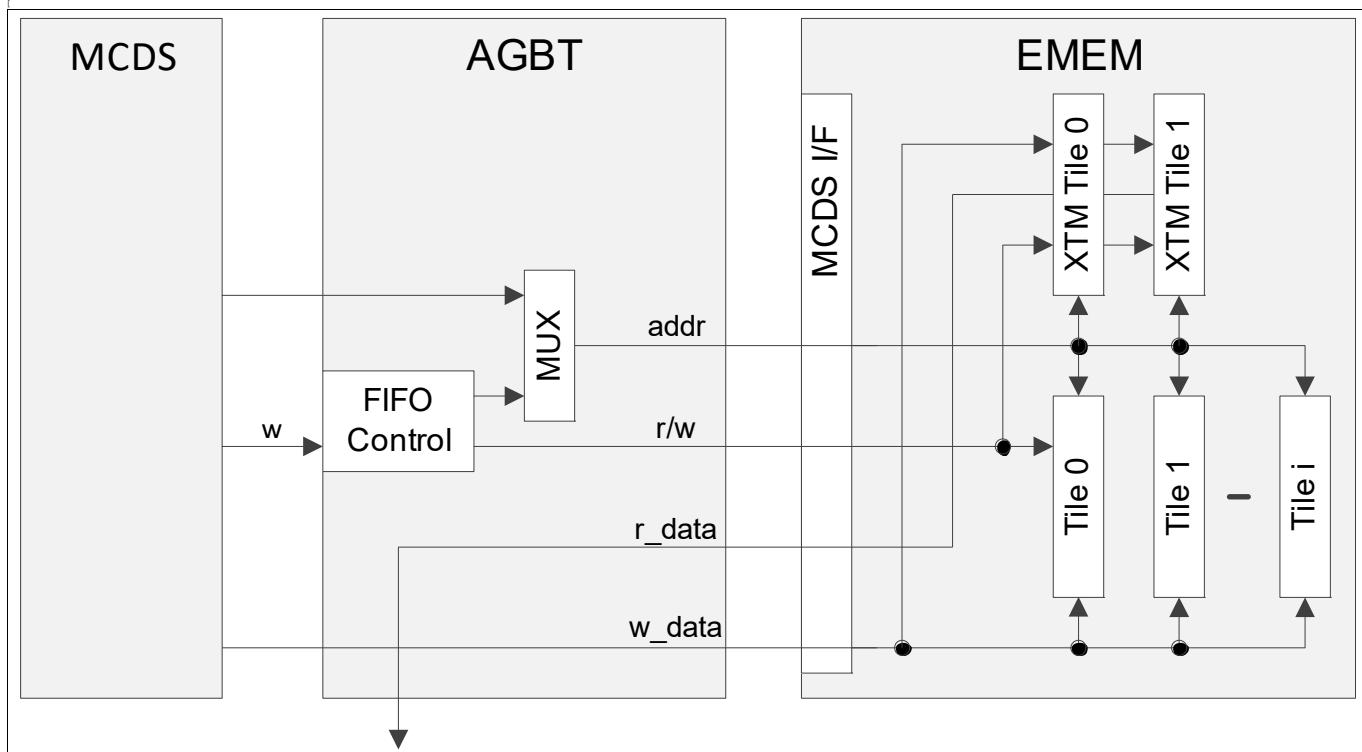


Figure 285 [[AGBT XTM0 and XTM1 read and write capability]]]]

### 23.3.4.2.2 [[Tool Mode]]

[The XTM is only accessible via the BBB fx slave interface.]]]]

## 23.3.5 [[EMEM Module SRAM]]

[An [EMEM Module] shall store data in the [EMEM Module] SRAM with a SECDED ECC generated checksum to provide protection against soft errors.]

### 23.3.5.1 [[SRAM Initialization]]

[After power-on the [EMEM Module] SRAM contents are random. To initialize the [EMEM Module] SRAM, software shall perform a write access to each 32-byte address aligned word line.]

#### [[SRI Slave Interface]]

[All access sizes may be used to initialize the [EMEM Module] SRAM as follows:]

- [[STDB, STDH, SDTW, SDTD and BTR2]]

## Extension Memory (EMEM)

- [[The [EMEM Module] shall perform an internal Read Modify Write (iRMW) to the SRAM. If the [EMEM Module] SRAM is un-initialized, the [EMEM Module] may generate a read phase ECC error preventing the write phase.]]
- [Prior to [EMEM Module] SRAM initialization, software shall configure Test Mode (MEMCON.ERRDIS = [1<sub>B</sub>]) to suppress read phase ECC errors.]
- [It is sufficient for software to perform one write access to any part of each 32-byte word line.]
- [As soon as [EMEM Module] SRAM initialization is complete, software shall configure Normal Mode (MEMCON.ERRDIS = [0<sub>B</sub>]) to enable the [Memory Integrity Check].]]
- [BTR4]
  - [[The [EMEM Module] shall not perform an iRMW.]]
  - [Software shall initialize the [EMEM Module] SRAM in Normal Mode (MEMCON.ERRDIS = [0<sub>B</sub>]).]]]

Note: [[For SRI accesses, an SDTD write access is the preferred method for initializing a 32-byte wordline.]]]

### [[BBB Slave Interface]]

[All access sizes may be used to initialise the [EMEM Module] SRAM. The [EMEM Module] ignores iRMW read phase ECC errors and always completes the write phase.]

Note: [[For BBB accesses, an SDTW write access is the preferred method for initializing a 32-byte wordline.]]]

### 23.3.5.2 [[Memory Integrity Check]]

[As soon as data is read from the SRAM, the EMEM module shall perform a [Memory Integrity Check] on the EMEM module SRAM read data to detect uncorrectable ECC errors.]

### 23.3.5.3 [[RAM Alarm]]

[If the EMEM detects an unexpected write to an [EMEM Module] SRAM, the EMEM shall trigger a RAM alarm.]

### 23.3.6 [[SRI Interface]]

[Each [EMEM Module] has its own SRI slave interface and configuration registers used to configure safety and security. When accessed via the SRI slave interface(s), the EMEM may be used for code execution, data storage or overlay memory. SRI accesses may be via cached (Segment [9<sub>H</sub>]) or non-cached (Segment [B<sub>H</sub>]) addresses.]

#### 23.3.6.1 [[Register Protection]]

[Each [EMEM Module] SRI slave interface defines independent write access enable<sup>1)</sup> to the [EMEM Module] registers.]

#### 23.3.6.2 [[Memory Protection]]

[An [EMEM Module] SRI slave interface controls access to its SRAM address space via a [MPU]. Each on chip resource with bus master capability has a unique master tag identifier that is used to identify the source of an on chip bus transaction. The master tag identifier based access protection is used to enable accesses to individual slave address ranges. Each [EMEM Module] shall have eight access protection regions of SRAM<sup>2)</sup>.]

[Each [Memory Protection] region shall be defined using six registers:]

1) See On-Chip System Connectivity for details.

2) Applies to SRAM and not SFR.

## Extension Memory (EMEM)

- [[The upper and lower 32-byte aligned address boundaries of the [Memory Protection] region shall be defined by:]
  - [[RGNLAX ( $i = 0-7$ ) shall define the lower address of the EMEM module [Memory Protection] region i.]]
  - [[RGNUAX ( $i = 0-7$ ) shall define the upper address of the EMEM module [Memory Protection] region i.]]
- [Two registers RGNACCENWAX ( $i = 0-7$ ) and RGNACCENWBX ( $i = 0-7$ ) shall select the on chip resources permitted to have write access to EMEM module [Memory Protection] region i.]
  - [[If Cerberus is present, then a Cerberus write access shall be enabled/disabled by [Memory Protection].]]
  - [[If HSM is present, then a HSM write access shall be enabled/disabled by [Memory Protection].]]
- [Two registers RGNACCENRAx ( $i = 0-7$ ) and RGNACCENRBx ( $i = 0-7$ ) shall select the on chip resources permitted to have read access to EMEM module [Memory Protection] region i.]
  - [[If Cerberus is present, then a Cerberus read access shall always be enabled.]]
  - [[If HSM is present, then a HSM read access shall always be enabled.]]

[After the application of a reset, all on chip resources shall have read and write access to all EMEM module [Memory Protection] regions.]

[If the [Memory Protection] defines overlapping EMEM module [Memory Protection] regions, then a read or write access made by an on chip resource access shall succeed if enabled by only one EMEM module [Memory Protection] region.]

### [[Reconfigure SRAM Protection]]

[If the EMEM module [Memory Protection] is to be re-configured, then the user should:]

- [[Wait for all pending EMEM module SRAM and SFR accesses to complete.]]
- [Write the EMEM module [Memory Protection] SRAM region registers to define updated configuration settings.]
- [Read the EMEM module [Memory Protection] SRAM region registers to check the configuration settings.]
- [As soon as the updated settings are confirmed, the system shall enable EMEM module accesses.]]]]

### 23.3.6.3 [[Memory Disabled]]

[If the [EMEM Module] SRAM is disabled (e.g. [EMEM Module] SRAM clock disabled), then an SRI access to the EMEM SRAM shall be terminated with an SRI bus error.]

### 23.3.6.4 [[True and Inverted Logic]]

[The [EMEM Module] SRI slave interface shall replicate the [Memory Integrity Check] using both true logic and inverted logic SECDED ECC decoders to eliminate common failure modes. If the true and inverted logic report a single error correction, the [EMEM Module] shall reconstruct the SRAM read data to be the original error free data.]]

### 23.3.6.5 [[Error Detection and Signalling]]

[Each [EMEM Module] shall detect several different classes of error (at [Table 1129]):]

- [[The [EMEM Module] shall trigger an alarm to the SMU.]]
- [The [EMEM Module] may set a status flag in the [EMEM Module] MEMCON register.]
- [The [EMEM Module] may complete the SRI access unless explicitly stated below.]]

## Extension Memory (EMEM)

Table 1129 [EMEM Module Alarms[ ]]

[Error Detection Event]	[Alarm]	[Status]
[[Access Enable Violation]]	[MPU Violation Alarm]	[ - ]
[[SRI Access Address Phase Error]]	[SRI Slave Address Phase Error Alarm]	[MEMCON.ADDERR]
[[SRI Write Access Data Phase Error]]	[SRI Slave Write Phase Error Alarm]	[MEMCON.DATAERR]
[[SRI Write Access to SRAM Error]]	[ECC Error Alarm]	[MEMCON.RMWERR]
[[SRI Read Access to SRAM Error]]	[ECC Error Alarm]	[ - ]
[[True and Inverted Logic Error]]	[EDC Read Phase Error Alarm]	[ - ]
[[Internal Data Transfer ECC Error]]	[ECC Error Alarm]	[MEMCON.INTERR]

[The EMEM and LMU share the same SMU alarms.]

## 23.3.6.5.1 [[Access Enable Violation]]

[If during a SRI access the [EMEM Module] detects either an [Register Protection] or [Memory Protection] violation, the [EMEM Module] shall trigger a [MPU Violation Alarm].]

- [[A SRI read access shall be terminated with a bus error.]
- [A SRI write access shall fail silently (no bus error). The [EMEM Module] shall not perform a write to SRAM.]]

## 23.3.6.5.2 [[SRI Access Address Phase Error]]

[If the [EMEM Module] detects an ECC error during the address phase of a SRI access, the [EMEM Module] shall record an address phase error (MEMCON.ADDERR = [1<sub>B</sub>]) and trigger a [SRI Slave Address Phase Error Alarm]. The SRI access shall terminate with an error.]

## 23.3.6.5.3 [[SRI Write Access Data Phase Error]]

[If the [EMEM Module] detects an ECC error during the data phase of a SRI write access, then the [EMEM Module] shall record a data phase error (MEMCON.DATAERR = [1<sub>B</sub>]) and trigger a [SRI Slave Write Phase Error Alarm].]

## 23.3.6.5.4 [[SRI Write Access to SRAM Error]]

[The [EMEM Module] SRAM is internally organized as a 256-bit SRAM. Any SRI write access of less than 256-bit shall require the [EMEM Module] to perform an internal Read-Modify-Write (iRMW) operation to correctly write the data and update the ECC checksum. If the [Memory Integrity Check] detects an uncorrectable ECC error during the read phase of the iRMW, the [EMEM Module] shall record a data error (MEMCON.RMWERR = [1<sub>B</sub>]) and trigger an [ECC Error Alarm].]]

## 23.3.6.5.5 [[SRI Read Access to SRAM Error]]

[If the [Memory Integrity Check] detects an uncorrectable ECC error during a SRI read access to the [EMEM Module] SRAM, the [EMEM Module] shall trigger an [ECC Error Alarm]. The SRI read access shall terminate with an SRI transaction ID error.]

[If ECC error disable is set (MEMCON.ERRDIS = [1<sub>B</sub>]), then the SRI read access shall terminate normally. The protection bit (MEMCON.PMIC) shall enable/disable configuration of the ECC error disabled bit (MEMCON.ERRDIS).]]

---

**Extension Memory (EMEM)****23.3.6.5.6 [[True and Inverted Logic Error]]**

[If the [EMEM Module] detects a difference in the [True and Inverted Logic] outputs during a [Memory Integrity Check], the [EMEM Module] shall trigger an [EDC Read Phase Error Alarm].]]

**23.3.6.5.7 [[Internal Data Transfer ECC Error]]**

[If the [EMEM Module] detects an uncorrected ECC error in an ECC protected internal data path, the [EMEM Module] shall record an internal error (MEMCON.INTERR = [1<sub>B</sub>]) and trigger an [ECC Error Alarm].]]]

## Extension Memory (EMEM)

## 23.3.6.6 [[Control Redundancy]]

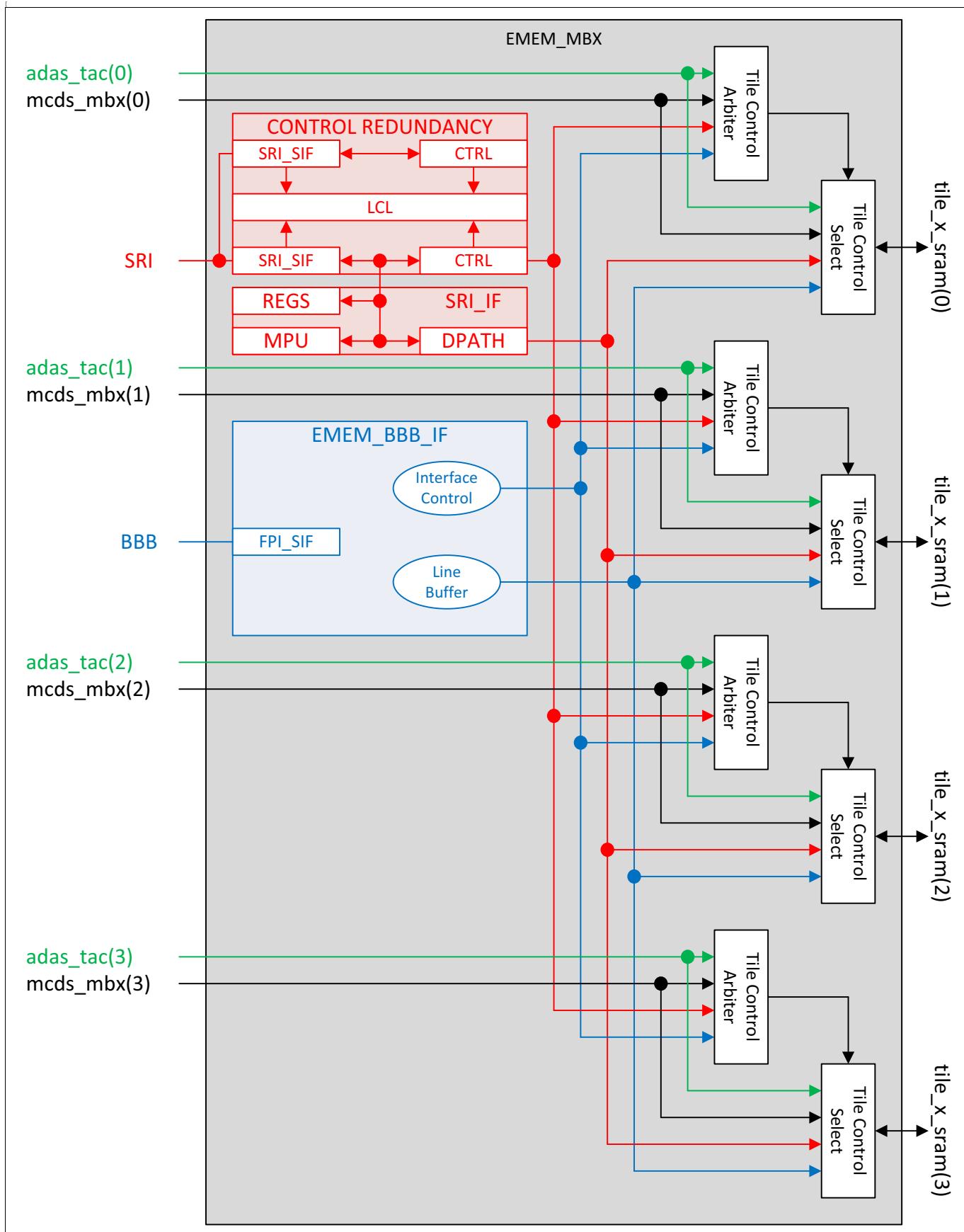


Figure 286 [Block Diagram of EMEM MBX]]

## Extension Memory (EMEM)

[The [EMEM Module] MBX block (at [Figure 286]) implements control redundancy to detect control faults during sub word SRI accesses to an [EMEM Module] register or SRAM. If the control redundancy detects a fault, the [EMEM Module] shall trigger an alarm to the SMU. The control redundancy shall be enabled after reset. The [EMEM Module] control redundancy and shall be disabled by software writing [01<sub>B</sub>] (OFF) to [EMEM Module] SCTRL.LSEN. The status of the [EMEM Module] control redundancy is reported by SCTRL.LSSTAT.]

### 23.3.6.6.1 [[Control Redundancy Test]]

[Software shall write [10<sub>B</sub>] to [EMEM Module] SCTRL.LSTST to test the [EMEM Module] control redundancy alarm. The control redundancy runs a background self test to periodically check the comparator functionality. If the self test fails, the [EMEM Module] shall trigger a SMU alarm.]

### 23.3.6.6.2 [[Consistency Check]]

[The [EMEM Module] control redundancy shall check the consistency of its own control state by maintaining all such information in redundant pairs of flip-flops where one flip-flop is the logical inverse of the other. If one of these pairs is in an inconsistent state (i.e. [00<sub>B</sub>] or [11<sub>B</sub>]), the [EMEM Module] shall trigger a SMU alarm.]]]

### 23.3.7 [[SEP Interface]]

[Multiple 256-bit [SEP] interfaces are used for dedicated data communication between SPU0/SPU1 and EMEM.]

[The EMEM shall give precedence to SEP accesses.]

#### 23.3.7.1 [[TC39xED SEP Accesses to EMEM Tiles]]

[The EMEM shall service the following multiple simultaneous SEP accesses to the same [EMEM Tile]:]

- [[One SEP write access and one SEP read access to the same EMEM tile.]]
- [[Two SEP read accesses to the same EMEM tile.]]

[SPU0 and SPU1 should only perform one SEP write access to one EMEM tile at any time. If SPU0 and SPU1 perform two or more simultaneous SEP write accesses to the same EMEM tile, the EMEM shall service the higher priority SEP write access and return an access error for the lower priority SEP write accesses. The order of precedence for SEP write accesses is:]

- [[sep\_wr0]]
- [sep\_wr1]
- [sep\_adc\_wr0]
- [sep\_adc\_wr1]]]

#### 23.3.7.2 [[TC35x SEP Accesses to EMEM Tiles]]

[The EMEM shall service the following multiple simultaneous SEP accesses to the same [EMEM Tile]:]

- [[If SPU0 and SPU1 are writing FFT results, two SEP write accesses to the same EMEM tile.]]
- [One SEP write access and one SEP read access to the same EMEM tile.]
- [[Two SEP read accesses to the same EMEM tile.]]

[If SPU0 and SPU1 are simultaneously writing FFT results to the same EMEM tile, the EMEM shall service SPU0 sep\_wr0 access and delay the servicing of the SPU1 sep\_wr1 access to the next clock cycle. If the EMEM is unable to service the SPU1 sep\_wr1 access, the EMEM will return an access error.]

[Else SPU0 and SPU1 should only perform one SEP write access to one EMEM tile at any time. If SPU0 and SPU1 perform two or more simultaneous SEP write accesses to the same EMEM tile, the EMEM shall service the higher

## Extension Memory (EMEM)

priority SEP write access and will return an access error for the lower priority SEP write accesses. The order of precedence for SEP write accesses is:]

- [[sep\_wr0]]
- [sep\_wr1]
- [sep\_adc\_wr0]
- [sep\_adc\_wr1]]]

### 23.3.7.3 [[TC3Ax SEP Accesses to EMEM Tiles]]

[The EMEM shall service the following multiple simultaneous SEP accesses to the same [EMEM Tile]:]

- [[If SPU0 and SPU1 are writing FFT results, two SEP write accesses to the same EMEM tile.]
- [One SEP write access and one SEP read access to the same EMEM tile.]
- [Two SEP read accesses to the same EMEM tile.]]

[If SPU0 and SPU1 are simultaneously writing FFT results to the same EMEM tile, the EMEM shall service SPU0 sep\_wr0 access and delay the servicing of the SPU1 sep\_wr1 access to the next clock cycle. If the EMEM is unable to service the SPU1 sep\_wr1 access, the EMEM will return an access error. If BITMGR attempts a write to the same tile when an SPU is writing then BITMGR shall be stalled until there is no other write access to that tile.]

[Else SPU0 and SPU1 should only perform one SEP write access to one EMEM tile at any time. If SPU0 and SPU1 perform two or more simultaneous SEP write accesses to the same EMEM tile, the EMEM shall service the higher priority SEP write access and will return an access error for the lower priority SEP write accesses. The order of precedence for SEP write accesses in this case is:]

- [[sep\_wr0]]
- [sep\_wr1]
- [sep\_adc\_wr0]
- [sep\_adc\_wr1]]]

[When there are two SEP read accesses to the same EMEM tile from the sep\_rd0 and sep\_rd1 interfaces the accesses shall be serviced using a round robin arbitration scheme. This gives a fair share of the read bandwidth between these two interfaces and avoids starvation. The sep\_rd2 is not included in the round robin and any read access by BITMGR to the same EMEM tile as a read access from sep\_rd0 or sep\_rd1 shall be stalled until there is no other read access to that EMEM tile.]]

### 23.3.7.4 [[TC33xED SEP Accesses to EMEM Tiles]]

[The EMEM shall service the following multiple simultaneous SEP accesses to the same [EMEM Tile]:]

- [[One SEP write access and one SEP read access to the same EMEM tile.]]

[If SPU0 performs two simultaneous SEP write accesses to the same EMEM tile, the EMEM shall service the higher priority SPU0 sep\_wr0 access and will return an access error for the lower priority SPU0 sep\_adc\_wr0 access.]]

### 23.3.7.5 [[SEP Error]]

[If SPU0/SPU1 accesses a disabled EMEM tile (via tile control), the EMEM shall signal an SEP error to SPU0/SPU1.]]

### 23.3.7.6 [[SEP ECC Error]]

[The SEP Interface protocol includes redundant error detection information (a combined address and data checksum) to allow the integrity checking of an [SEP Write Access] or [SEP Read Access]. ]

## Extension Memory (EMEM)

### 23.3.7.6.1 [[SEP Write Access]]

[If an SPU performs an SEP write access to EMEM, the EMEM shall check the integrity of the SEP write access.]

- [[The EMEM shall analyze the SEP control signal differential pairs.]
  - [[If the EMEM detects a differential error, the EMEM shall trigger an SEP alarm to the SMU.]]
- [The EMEM shall check the integrity of the access by analyzing the combined address and data checksum.]
  - [[If the EMEM detects an integrity error, the EMEM shall trigger an SEP alarm to the SMU.]]
  - [The EMEM shall service the SEP write access by performing a write to SRAM.]]]]]

### 23.3.7.6.2 [[SEP Read Access]]

[If an SPU performs an SEP read access to EMEM, the EMEM shall check the integrity of the SEP read access.]

- [[The EMEM shall analyze the SEP control signal differential pairs.]
  - [[If the EMEM detects a differential error, the EMEM shall trigger an SEP alarm to the SMU.]]
- [The EMEM shall calculate a combined address and data checksum for propagation from EMEM to SPU0/1.]]]]]

### 23.3.7.7 [[SPU Full Lockstep]]

[If SPU0 and SPU1 are configured for full lockstep operation, the EMEM shall function as follows:]

- [[SPU1 SEP write accesses to EMEM shall fail silently.]]
- [SPU1 SEP read accesses to EMEM shall be ignored.]
  - [[The EMEM shall return the SPU0 read data to SPU1.]]]]]

### 23.3.8 [[Reset Control]]

[The EMEM has two types of reset control:]

- [[The **[EMEM Module]** SRI interface shall be reset by application reset.]]
- [The remainder of EMEM shall be reset by EEC reset.]]

[If OCDS is disabled, the ECC reset is tied to the application reset. ]]

### 23.3.9 [[Clock Control]]

[The EMEM has two types of clock control registers:]

- [[The **[EMEM Core]** clock control register shall enable and disable the internal  $f_{[BBB]}$  clock in order to control the power consumption.]
  - [[If the **[EMEM Core]** clock control register disables the clock, then software shall be able to write to the **[EMEM Core]** clock control register to enable the internal EMEM clock.]]
- [Each **[EMEM Module]** has an independent clock control register to enable and disable the internal  $f_{[SRI]}$  clock to the **[EMEM Module]** SRI slave interface.]
  - [[If the **[EMEM Module]** clock control register disables the clock, then the **[EMEM Module]** shall service SRI accesses to the **[EMEM Module]** registers. SRI accesses to EMEM tiles shall not be serviced. ]
  - [The **[EMEM Module]** shall service and SRI access to an **[EMEM Tile]** when the **[EMEM Module]** clock is enabled and the **[EMEM Tile]** is configured for use via the **[EMEM Core]** TILECONFIG and TILEECC registers. ]]]]]]

## 23.4 [[Registers]]

[This section describes the registers of the EMEM module.]

## Extension Memory (EMEM)

### 23.4.1 [[EMEM Core Register Description]]

[[The [EMEM Core] registers shall be accessed via the BBB fc BPI SFF. The register description shows the maximum configuration. The number of active control and status bits shall depend on the EMEM configuration.]

[[**Table 1130 [Register Address Space - EMEM[ ]]**]]

[Module]	[Base Address]	[End Address]	[Note]
fc	[00000000 <sub>H</sub> ]	[0000FFFF <sub>H</sub> ]	BPI SFF (access to EMEM core registers)

[[**Table 1131 [Register Overview - EMEM (ascending Offset Address)[ ]]**]]

[Short Name]	[Long Name]	[Offset Address ]	[Access Mode]		[Reset]	[Page Number]
			[Read]	[Write]		
[CLC]	[EMEM Core Clock Control Register]	[[0000 <sub>H</sub> ]]	[U,SV]	[SV,E,P]	[[EEC Reset]]	[[18]]
[ID]	[EMEM Core Module Identification Register]	[[0008 <sub>H</sub> ]]	[U,SV]	[BE]	[[EEC Reset]]	[[19]]
[TILECONFIG]	[EMEM Core Tile Configuration Register]	[[0020 <sub>H</sub> ]]	[U,SV]	[U,SV,P]	[[EEC Reset]]	[[20]]
[TILECC]	[EMEM Core Tile Control Common Memory Register]	[[0024 <sub>H</sub> ]]	[U,SV]	[U,SV,P]	[[EEC Reset]]	[[20]]
[TILECT]	[EMEM Core Tile Control Trace Memory Register]	[[0028 <sub>H</sub> ]]	[U,SV]	[U,SV,P]	[[EEC Reset]]	[[21]]
[TILESTATE]	[EMEM Core Tile Status Register]	[[002C <sub>H</sub> ]]	[U,SV]	[BE]	[[EEC Reset]]	[[22]]
[SBRCTR]	[EMEM Core Standby RAM Control Register]	[[0034 <sub>H</sub> ]]	[U,SV]	[U,SV,P]	[[EEC Reset]]	[[22]]
[ACCEN1]	[EMEM Core Access Enable Register 1]	[[00F8 <sub>H</sub> ]]	[U,SV]	[BE]	[[EEC Reset]]	[[23]]
[ACCENO]	[EMEM Core Access Enable Register 0]	[[00FC <sub>H</sub> ]]	[U,SV]	[SV,SE]	[[EEC Reset]]	[[24]]

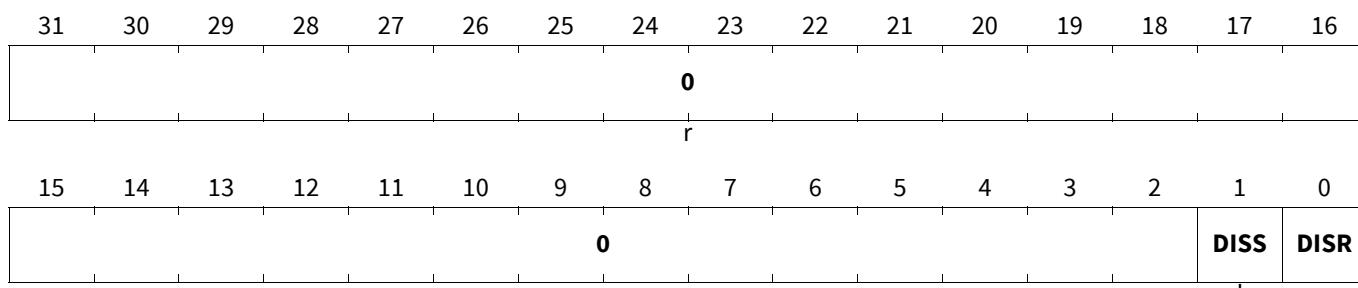
[[If the [EMEM Core] clock is disabled, all [EMEM Core] register accesses (except CLC) shall BE.]]

#### [[[[EMEM Core Clock Control Register]]]

[[The EMEM core clock control register shall enable and disable the internal EMEM clock.]

**Extension Memory (EMEM)**

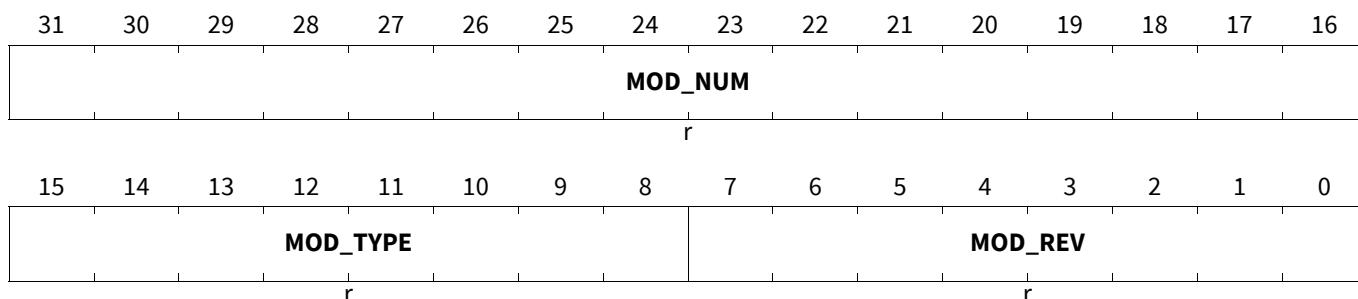
[CLC]

**[EMEM Core Clock Control Register]**(0000<sub>H</sub>)[EEC Reset] Value: 0000 0003<sub>H</sub>

[Field]	[Bits]	[Type]	[Description]
[DISR]	[0]	[rw]	<b>[Module Disable Request Bit]</b> [Used for enable/disable control of the EMEM] [[[0 <sub>B</sub> ]][EMEM disable is not requested. ]] [[[1 <sub>B</sub> ]][EMEM disable is requested. ]]
[DISS]	[1]	[rh]	<b>[Module Disable Status Bit]</b> [Module Disable Status Bit] [[[0 <sub>B</sub> ]][EMEM is enabled. ]] [[[1 <sub>B</sub> ]][EMEM is disabled.]]
[0]	[31:2]	[r]	<b>[Reserved]</b> [Read as 0; should be written with 0.]

**[[[EMEM Core Module Identification Register]]]**

[ID]

**[EMEM Core Module Identification Register]**(0008<sub>H</sub>)[EEC Reset] Value: 00E0 C0XX<sub>H</sub>

[Field]	[Bits]	[Type]	[Description]
[MOD_REV]	[7:0]	[r]	<b>[Module Revision Number]</b> [This bit field defines e module revision number.] [See EMEM Design Specification for MOD_REV value.]
[MOD_TYPE]	[15:8]	[r]	<b>[Module Type]</b> [The bit field is set to C0 <sub>H</sub> which defines the module as a 32-bit module.]
[MOD_NUM]	[31:16]	[r]	<b>[Module Number]</b> [This bit field defines a module identification number.]

**Extension Memory (EMEM)****[[[EMEM Core Tile Configuration Register]]****[TILECONFIG]**

**[EMEM Core Tile Configuration Register (0020<sub>H</sub>)]**      **[EEC Reset] Value: 5555 5555<sub>H</sub>]**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
XCM7	XCM6	XCM5	XCM4	XCM3	XCM2	XCM1	XCM0								
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
TCM7	TCM6	TCM5	TCM4	TCM3	TCM2	TCM1	TCM0								
w	w	w	w	w	w	w	w								

[Field]	[Bits]	[Type]	[Description]
[[TCMx] (x=0-7)]	[2*x+1:2*x]	[w]	<b>[TCM Tile x Assignment Change]</b>  <i>Note:</i> [[[00 <sub>B</sub> ]]][Assign EMEM Tile to Common Memory.] ] [[[01 <sub>B</sub> ]]][No change of EMEM Tile assignment.] ] [[[10 <sub>B</sub> ]]][Assign EMEM Tile to Trace Memory.] ] [[[11 <sub>B</sub> ]]][Set EMEM Tile to Unused Mode.] ]
[[XCMx] (x=0-7)]	[2*x+17:2*x+16]	[w]	<b>[XCM Tile x Assignment Change]</b>  <i>Note:</i> [[[00 <sub>B</sub> ]]][Assign EMEM Tile to Common Memory.] ] [[[01 <sub>B</sub> ]]][No change of EMEM Tile assignment.] ] [[[10 <sub>B</sub> ]]][Reserved.] ] [[[11 <sub>B</sub> ]]][Set EMEM Tile to Unused Mode.] ]

**[[[EMEM Core Tile Control Common Memory Register]]****[TILECC]**

**[EMEM Core Tile Control Common Memory Register(0024<sub>H</sub>)]**

**[EEC Reset] Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								XCM15	XCM14	XCM13	XCM12	XCM11	XCM10	XCM9	XCM8
r								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XCM7	XCM6	XCM5	XCM4	XCM3	XCM2	XCM1	XCM0	TCM7	TCM6	TCM5	TCM4	TCM3	TCM2	TCM1	TCM0
rw	rw	rw	rw	rw	rw	rw	rw								

## Extension Memory (EMEM)

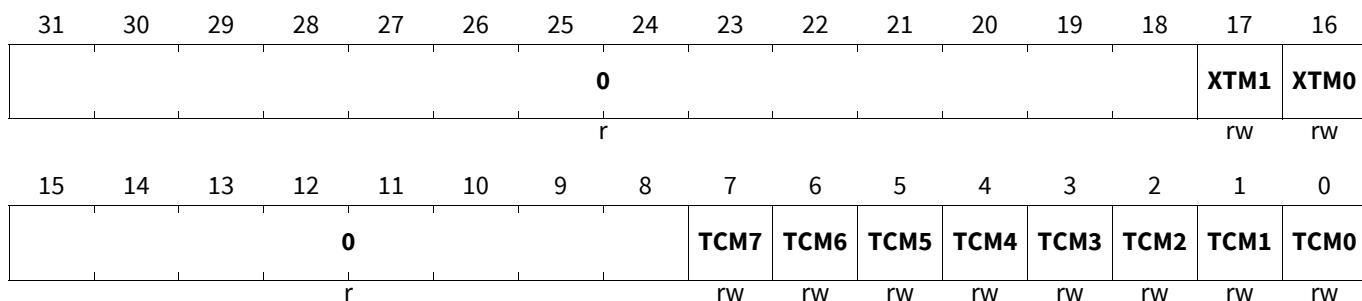
[Field]	[Bits]	[Type]	[Description]
[[TCMx] (x=0-7)]	[x]	[rw]	<p><b>Common Memory TCM Tile x Control</b></p> <p>[No effect when the EMEM tile is not present, assigned to Trace Memory or in Unused Mode.]</p> <p>[[[0<sub>B</sub>] ][Application/ADAS Mode ]]</p> <p>[[[1<sub>B</sub>] ][Tool Mode ]]</p>
[[XCMx] (x=0-15)]	[x+8]	[rw]	<p><b>Common Memory XCM Tile x Control</b></p> <p>[No effect when the EMEM tile is not present, assigned to Trace Memory or in Unused Mode.]</p> <p>[[[0<sub>B</sub>] ][Application/ADAS Mode ]]</p> <p>[[[1<sub>B</sub>] ][Tool Mode ]]</p>
[0]	[31:24]	[r]	<p><b>Reserved</b></p> <p>[Read as 0, should be written with 0.]</p>

## [[[EMEM Core Tile Control Trace Memory Register]]]

[TILECT]

## [EMEM Core Tile Control Trace Memory Register(0028[\_H])]

[EEC Reset] Value: 0000 0000[H]



[Field]	[Bits]	[Type]	[Description]
[[TCMx] (x=0-7)]	[x]	[rw]	<p><b>[Trace Memory TCM Tile x Control Bit]</b></p> <p>[No effect when the EMEM tile is not present, assigned to Common Memory or in Unused Mode.]</p> <p>[[[0<sub>B</sub>] ][MCDS Mode ]]</p> <p>[[[1<sub>B</sub>] ][Tool Mode ]]</p>
[[XTMx] (x=0-1)]	[x+16]	[rw]	<p><b>[Trace Memory XTM Tile x Control Bit]</b></p> <p>[If both the XTM and the associated TCM Tile are set to MCDS Mode (reset value) the MCDS output shall be to the XTM Tile.]</p> <p>[[[0<sub>B</sub>] ][MCDS Mode ]]</p> <p>[[[1<sub>B</sub>] ][Tool Mode (only BBB access) ]]</p>
[0]	[15:8, 31:18]	[r]	<p><b>[Reserved]</b></p> <p>[Read as 0; should be written with 0.]</p>

## Extension Memory (EMEM)

### [[[EMEM Core Tile Status Register]]]

[[TILESTATE]]

[EMEM Core Tile Status Register]

(002C<sub>H</sub>)][EEC Reset] Value: FFFF FFFF<sub>H</sub>]]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
XCM7	XCM6	XCM5	XCM4	XCM3	XCM2	XCM1	XCM0								
rh	rh	rh	rh	rh	rh	rh	rh	rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCM7	TCM6	TCM5	TCM4	TCM3	TCM2	TCM1	TCM0								
rh	rh	rh	rh	rh	rh	rh	rh	rh							

[Field]	[Bits]	[Type]	[Description]
[[TCMx] (x=0-7)]	[2*x+1:2*x]	[rh]	<b>[Assignment of TCM Tile x]</b> [[[00 <sub>B</sub> ]][Common Memory ]] [[[01 <sub>B</sub> ]][Reserved ]] [[[10 <sub>B</sub> ]][Trace Memory ]] [[[11 <sub>B</sub> ]][Unused Mode or Tile not present ]]
[[XCMx] (x=0-7)]	[2*x+17:2*x+16]	[rh]	<b>[Assignment of XCM Tile x]</b> [[[00 <sub>B</sub> ]][Common Memory ]] [[[01 <sub>B</sub> ]][Reserved ]] [[[10 <sub>B</sub> ]][Reserved ]] [[[11 <sub>B</sub> ]][Unused Mode or Tile not present ]]

### [[[EMEM Core Standby RAM Control Register]]]

[[SBRCTR]]

[EMEM Core Standby RAM Control Register]

(0034<sub>H</sub>)][EEC Reset] Value: 0000 0000<sub>H</sub>]]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
															STBP ON
															rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															STBLO CK

[Field]	[Bits]	[Type]	[Description]
[STBLOCK]	[0]	[rh]	<b>[Standby Lock Flag]</b> [EMEM Mode] [[[0 <sub>B</sub> ]][Locked Mode. ]] [[[1 <sub>B</sub> ]][Unlocked Mode. ]]

## Extension Memory (EMEM)

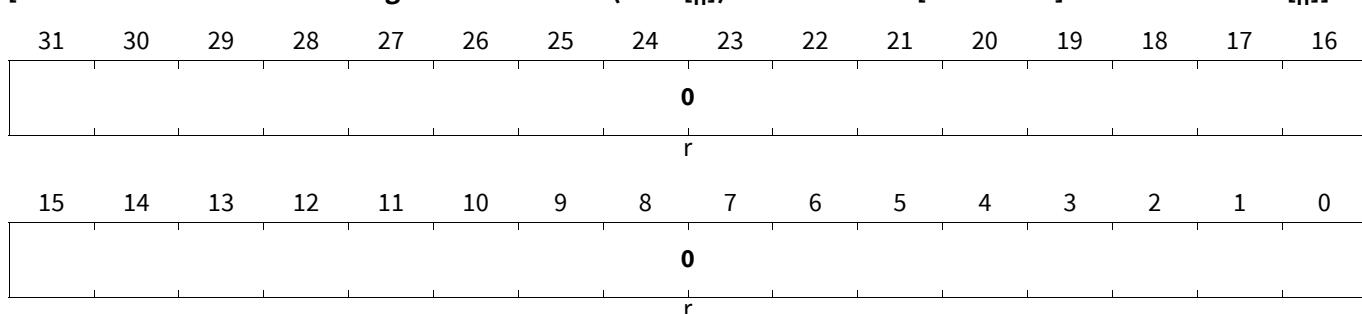
[Field]	[Bits]	[Type]	[Description]
[STBULK]	[3:1]	[w]	<p><b>[Unlock Standby Lock Flag]</b>            [In order to transition the EMEM (including XCM and XTM) from Standby Locked Mode to Unlocked Mode in three consecutive<sup>1)</sup> write cycles the following patterns have to be written into this bit field:]</p> <ul style="list-style-type: none"> <li>• [[[001<sub>B</sub>]]]</li> <li>• [[011<sub>B</sub>]]</li> <li>• [[111<sub>B</sub>]]]</li> </ul> <p>[At the same time 0 has to be written always into bit field STBSLK. If any of STBSLK is set when writing a non-zero pattern to STBULK, this is treated as invalid pattern and the EMEM in Standby Locked Mode will not transition to Unlocked Mode. Reading this bit field always will return 0s.]</p>
[STBSLK]	[7:4]	[w]	<p><b>[Set Standby Lock Flag]</b>            [In order to lock the Extension Memory including XCM and XTM in operating mode [1001<sub>B</sub>] has to be written into this bit field.]</p> <p>[At the same time 0 has to be written into the bit field STBULK. If any of bits STBULK is set when writing [1001<sub>B</sub>] to STBSLK, this is treated as invalid pattern and the Lock Flag is not set.]</p> <p>[Reading this bit field always will return 0s]</p>
[STBPON]	[16]	[rh]	<p><b>[Standby Power On]</b>            [Status flag for EMEM Standby Power Domain. The standby power is used to provide power to the EMEM tiles. It should be available before the EMEM is switched to Unlocked Mode.]</p> <p>[[[0<sub>B</sub>]]][EMEM Standby Power Domain is not available or out of limits. ]]          [[[1<sub>B</sub>]]][EMEM Standby Power Domain is available and within limits. ]]</p>
[0]	[15:8, [31:17]	[r]	<p><b>[Reserved]</b>            [Read as 0; should be written with 0.]</p>

1) Intermediate read or write cycles not accessing [SBRCTR] are allowed.

## [[[EMEM Core Access Enable Register 1]]]

## [[ACCEN1]]

[EMEM Core Access Enable Register 1] (00F8[<sub>H</sub>]) [EEC Reset] Value: 0000 0000[<sub>H</sub>]]



[Field]	[Bits]	[Type]	[Description]
[0]	[31:0]	[r]	<p><b>[Reserved]</b>            [Read as 0; should be written with 0.]</p>

## Extension Memory (EMEM)

### [[[EMEM Core Access Enable Register 0]

[[ACCEN0]]

[EMEM Core Access Enable Register 0]

(00FC<sub>H</sub>)

[EEC Reset] Value: FFFF FFFF<sub>H</sub>]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

[Field]	[Bits]	[Type]	[Description]
[[ENq] (q=0-31)]	[q]	[rw]	<p><b>Access Enable for Master TAG ID q]</b>            [This bit enables write access to the module kernel addresses for transactions with the Master TAG ID q]            [[[0<sub>B</sub> ]][Write access will not be executed ]]            [[[1<sub>B</sub> ]][Write access will be executed ]]</p>

## Extension Memory (EMEM)

## 23.4.2 [[EMEM Module Register Description]]

[[The [EMEM Module] registers shall be accessed via the EMEM Module SRI slave interface.]

Table 1132 [Register Address Space - EMEM\_MPU[ ]]

[Module]	[Base Address]	[End Address]	[Note]
s0	[00000000 <sub>H</sub> ]	[0000FFFF <sub>H</sub> ]	SRI slave interface 0 (access to EMEM module registers)

Table 1133 [Register Overview - EMEM\_MPU (ascending Offset Address)[ ]]

[Short Name]	[Long Name]	[Offset Address ]	[Access Mode]		[Reset]	[Page Number]
			[Read]	[Write]		
[CLC]	[EMEM Module Clock Control Register]	[[00000 <sub>H</sub> ]]	[SV]	[SV,E,P]	[[Application Reset]]	[[26]]
[MODID]	[EMEM Module ID Register]	[[00008 <sub>H</sub> ]]	[SV]	[R]	[[Application Reset]]	[[26]]
[ACCEN0]	[EMEM Module Access Enable Register 0]	[[00010 <sub>H</sub> ]]	[SV]	[SV,SE]	[[Application Reset]]	[[27]]
[ACCEN1]	[EMEM Module Access Enable Register 1]	[[00014 <sub>H</sub> ]]	[SV]	[SV,SE]	[[Application Reset]]	[[27]]
[MEMCON]	[EMEM Module Memory Control Register]	[[00020 <sub>H</sub> ]]	[SV]	[SV,E,P]	[[Application Reset]]	[[28]]
[SCTRL]	[EMEM Module Safety Control Register]	[[00024 <sub>H</sub> ]]	[SV]	[SV,SE,P]	[[Application Reset]]	[[29]]
[RGNLAI]	[EMEM Module Region i Lower Address Register]	[[00050 <sub>H</sub> ] +i*[10 <sub>H</sub> ]]	[SV]	[SV,SE,P]	[[Application Reset]]	[[30]]
[RGNUAI]	[EMEM Module Region i Upper Address Register]	[[00054 <sub>H</sub> ] +i*[10 <sub>H</sub> ]]	[SV]	[SV,SE,P]	[[Application Reset]]	[[30]]
[RGNACCENWAI]	[EMEM Module Region i Write Access Enable Register 0]	[[00058 <sub>H</sub> ] +i*[10 <sub>H</sub> ]]	[SV]	[SV,SE,P]	[[Application Reset]]	[[31]]
[RGNACCENWBi]	[EMEM Module Region i Write Access Enable Register 1]	[[0005C <sub>H</sub> ] +i*[10 <sub>H</sub> ]]	[SV]	[SV,SE,P]	[[Application Reset]]	[[32]]
[RGNACCENRAi]	[EMEM Module Region i Read Access Enable Register 0]	[[000D8 <sub>H</sub> ] +i*[10 <sub>H</sub> ]]	[SV]	[SV,SE,P]	[[Application Reset]]	[[32]]
[RGNACCENRBi]	[EMEM Module Region i Read Access Enable Register 1]	[[000DC <sub>H</sub> ] +i*[10 <sub>H</sub> ]]	[SV]	[SV,SE,P]	[[Application Reset]]	[[33]]

## Extension Memory (EMEM)

### 23.4.2.1 [[EMEM Module General Registers]]

#### [[[EMEM Module Clock Control Register]]]

[The EMEM module clock control register shall enable and disable the clock to the SRI slave interface.]

[[CLC]]

[[[EMEM Module Clock Control Register]]] (00000[H]) [Application Reset] Value: 0000 0000[H]															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														DISS	DISR
rh														rw	
r															

[Field]	[Bits]	[Type]	[Description]
[DISR]	[0]	[rw]	<b>[Module Disable Request Bit]</b> [Used for enable/disable control of the EMEM module SRI slave interface.] [[[0 <sub>B</sub> ]] [ EMEM module disable is not requested. ]] [[[1 <sub>B</sub> ]] [ EMEM module disable is requested. ]]
[DISS]	[1]	[rh]	<b>[Module Disable Status Bit]</b> [Bit indicates the current state of the EMEM module SRI slave interface.] [[[0 <sub>B</sub> ]] [ EMEM module is enabled. ]] [[[1 <sub>B</sub> ]] [ EMEM module is disabled. ]]
[0]	[31:2]	[r]	<b>[Reserved]</b> [Read as 0; should be written with 0.]

#### [[[EMEM Module ID Register]]]

[[MODID]]

[[[EMEM Module ID Register]]] (00008[H]) [Application Reset] Value: 0088 C0XX[H]															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUMBER															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

[Field]	[Bits]	[Type]	[Description]
[MOD_REV]	[7:0]	[r]	<b>[Module Revision Number]</b> [This bit field defines the module revision number.]

## Extension Memory (EMEM)

[Field]	[Bits]	[Type]	[Description]
[MOD_TYPE]	[15:8]	[r]	<b>[Module Type]</b> [The bit field is set to C0 <sub>H</sub> which defines the module as a 32-bit module.]
[MOD_NUMBE R]	[31:16]	[r]	<b>[Module Number Value]</b> [This bit field defines a module identification number.]

## [[[EMEM Module Access Enable Register 0]]]

[[ACCEN0]]  
**[EMEM Module Access Enable Register 0]** (00010<sub>H</sub>) **[Application Reset] Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

[[[ACCEN1]]]

[Field]	[Bits]	[Type]	[Description]
[[ENn] (n=0-31)]	[n]	[rw]	<b>[Access Enable for Master TAG ID n]</b> [This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n] [[[0 <sub>B</sub> ]]][Write access will not be executed. ]] [[[1 <sub>B</sub> ]]][Write access will be executed. ]]

## [[[EMEM Module Access Enable Register 1]]]

[[ACCEN1]]  
**[EMEM Module Access Enable Register 1]** (00014<sub>H</sub>) **[Application Reset] Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw															

[[[ACCEN2]]]

[Field]	[Bits]	[Type]	[Description]
[[ENn] (n=32-63)]	[n-32]	[rw]	<b>[Access Enable for Master TAG ID n]</b> [This bit enables write access to the module kernel addresses for transactions with Master TAG ID n] [[[0 <sub>B</sub> ]]][Write access will not be executed. ]] [[[1 <sub>B</sub> ]]][Write access will be executed. ]]

## Extension Memory (EMEM)

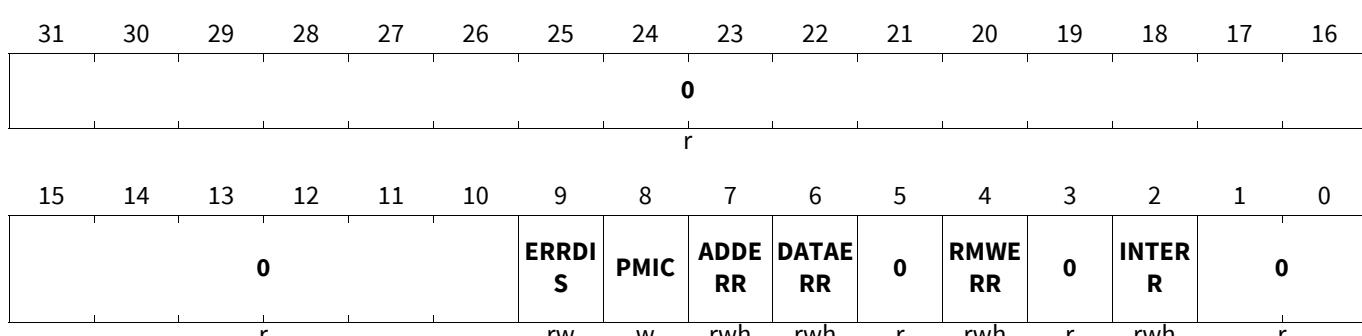
### [[[EMEM Module Memory Control Register]]]

[Controls the memory integrity error checking and error signalling to the SMU.]

[Note: LDMST or SWAPMSK.W should be used only with bit mask enabled for all 'rwh' bits in the addressed register.]

#### [MEMCON]

[EMEM Module Memory Control Register] (00020<sub>H</sub>) [Application Reset] Value: 0000 0000<sub>H</sub>



[Field]	[Bits]	[Type]	[Description]
[INTERR]	[2]	[rwh]	<b>[Internal ECC Error]</b> [Flag set when the EMEM module logic detects an uncorrectable ECC error during the transfer of data between the SRI interface and the RAM. This bit is cleared by writing 0 <sub>B</sub> but cannot be set by software.] [[[0 <sub>B</sub> ]][No error has occurred.]] [[[1 <sub>B</sub> ]][An error has been observed during a RAM access.]]
[RMWERR]	[4]	[rwh]	<b>[Internal Read Modify Write Error]</b> [Flag set when the EMEM module logic detects an uncorrectable ECC error during the read phase of an internal RMW access to RAM. This bit is cleared by writing 0 <sub>B</sub> but cannot be set by software.] [[[0 <sub>B</sub> ]][No error has occurred.]] [[[1 <sub>B</sub> ]][An error has been observed during an internal RMW operation.]]
[DATAERR]	[6]	[rwh]	<b>[SRI Data Phase ECC Error]</b> [Flag set when the EMEM module SRI slave interface detects an uncorrectable ECC error during the data phase of an on chip bus access to RAM. This bit is cleared by writing 0 <sub>B</sub> but cannot be set by software.] [[[0 <sub>B</sub> ]][No error has occurred.]] [[[1 <sub>B</sub> ]][An error has occurred.]]
[ADDERR]	[7]	[rwh]	<b>[SRI Address Phase ECC Error]</b> [Flag set by hardware when the SRI interface detects an ECC error in the address phase of an incoming transaction. This bit is cleared by writing 0 <sub>B</sub> but cannot be set by software.] [[[0 <sub>B</sub> ]][No error has occurred.]] [[[1 <sub>B</sub> ]][An error has occurred.]]
[PMIC]	[8]	[w]	<b>[Protection Bit for Memory Integrity Control Bit]</b> [Will always return 0 <sub>B</sub> when read.] [[[0 <sub>B</sub> ]][ERRDIS remains unchanged after a write to MEMCON.]] [[[1 <sub>B</sub> ]][ERRDIS will be updated by the current write to MEMCON.]]

## Extension Memory (EMEM)

[Field]	[Bits]	[Type]	[Description]
[ERRDIS]	[9]	[rw]	<p><b>[ECC Error Disable]</b>            Controls the reporting of bus errors when the slave interface detects an uncorrectable ECC error during an on chip bus read access to RAM.]</p> <p>[[[0<sub>B</sub>]] [Normal operation. Bus error is reported. ]]            [[[1<sub>B</sub>]] [Test mode. No bus error is reported. ]]</p>
[0]	[1:0, [3, [5, [31:10]	[r]	<p><b>[Reserved]</b>            [Read as 0; should be written as 0.]</p>

### [[[EMEM Module Safety Control Register]]]

[The safety control register shall provide a means of injecting errors into the data integrity checking logic.]

■

#### [SCTRL]

[EMEM Module Safety Control Register] (00024<sub>H</sub>) [Application Reset] Value: 0002 0600<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

0															
	r														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				LSTST		LSEN				0			GEC		GED
r				rw		rw				r			w		w

[Field]	[Bits]	[Type]	[Description]
[GED]	[0]	[w]	<p><b>[Generate Error in ECC for Data Protection]</b>            [The data paths between the SRAM and the EMEM bus interface are protected by ECC logic. This bit is used to inject an error into the next access so that the SMU alarm shall be tested. Reading this bit always returns [0<sub>B</sub>]. Writing works as follows:]</p> <p>[[[0<sub>B</sub>]] [No effect. ]]            [[[1<sub>B</sub>]] [Inject error during next RAM access. ]]</p>
[GEC]	[1]	[w]	<p><b>[Generate Error in ECC for Error Correction]</b>            [The data read from the SRAM is corrected by ECC logic. This ECC logic is duplicated so the functionality can be checked. This bit is used to inject an error into the next access so that the SMU alarm can be tested. Reading this bit always returns [0<sub>B</sub>]. Writing works as follows:]</p> <p>[[[0<sub>B</sub>]] [No effect. ]]            [[[1<sub>B</sub>]] [Inject error during next RAM read access. ]]</p>
[LSEN]	[9:8]	[rw]	<p><b>[Lockstep Enable]</b>            [Control of comparators checking the duplicated logic area for errors.]</p> <p>[[[00<sub>B</sub>]] [RES0][, Invalid]]            [[[01<sub>B</sub>]] [OFF][, Lockstep Off]]            [[[10<sub>B</sub>]] [ON][, Lockstep On]]            [[[11<sub>B</sub>]] [RES3][, Invalid]]]</p>

## Extension Memory (EMEM)

[Field]	[Bits]	[Type]	[Description]
[LSTST]	[11:10]	[rw]	<p><b>[Lockstep Test]</b>  [Setting this bitfield will inject an error into the comparators checking the duplicated logic area. This will allow the correct operation of the SMU alarm to be verified. An error will continue to be injected until this field is reset.]</p> <p>[[[00<sub>B</sub>]][RES0][, Invalid]]  [[[01<sub>B</sub>]][OFF][, No error injected]]  [[[10<sub>B</sub>]][ON][, Error injected]]  [[[11<sub>B</sub>]][RES3][, Invalid]]</p>
[LSSTAT]	[17:16]	[rh]	<p><b>[Lockstep Status]</b>  [Reports the status of the comparators.]</p> <p>[[[00<sub>B</sub>]][RES0][, Invalid]]  [[[01<sub>B</sub>]][OFF][, Lockstep is Off]]  [[[10<sub>B</sub>]][ON][, Lockstep is On]]  [[[11<sub>B</sub>]][RES3][, Invalid]]</p>
[0]	[7:2] [15:12] [31:18]	[r]	<p><b>[Reserved]</b>  [Read as 0; should be written as 0.]</p>

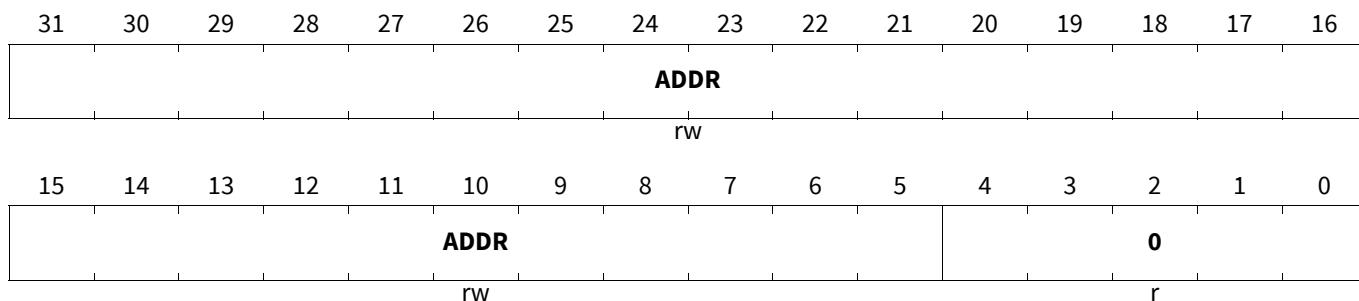
### 23.4.2.2 [[EMEM Module SRAM Protection Registers]]

#### [[[EMEM Module Region i Lower Address Register]]

[RGNLAI shall store the lower address of the EMEM module SRAM protection region i.]

[[[RGNLAI] (i=0-7)]

[EMEM Module Region i Lower Address Register(00050<sub>H</sub>+i\*10<sub>H</sub>)][Application Reset] Value: 0000 0000<sub>H</sub>]



[Field]	[Bits]	[Type]	[Description]
[ADDR]	[31:5]	[rw]	<p><b>[Region Lower Address]</b>  [Bits 31 to 5 of the address which is the lower bound of the defined memory region.]</p>
[0]	[4:0]	[r]	<p><b>[Reserved]</b>  [Read as 0; should be written with 0.]</p>

#### [[[EMEM Module Region i Upper Address Register]]

[RGNUI shall store the upper address of the EMEM module SRAM protection region i.]

## Extension Memory (EMEM)

[[RGNUAi] (i=0-7)]

[EMEM Module Region i Upper Address Register(00054[\_H]+i\*10[\_H])[Application Reset] Value: FFFF FFEO[\_H]]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR															
rw															

[Field]	[Bits]	[Type]	[Description]
[ADDR]	[31:5]	[rw]	<b>[Region Upper Address]</b> [Bits 31 to 5 of the address which is the upper bound of the defined memory region.]
[0]	[4:0]	[r]	<b>[Reserved]</b> [Read as 0; should be written with 0.]

[[[EMEM Module Region i Write Access Enable Register 0]

[[RGNACCENWAI] (i=0-7)]

[EMEM Module Region i Write Access Enable Register 0(00058[\_H]+i\*10[\_H])[Application Reset] Value: FFFF FFFF[\_H]]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

[Field]	[Bits]	[Type]	[Description]
[[ENn] (n=0-31)]	[n]	[rw]	<b>[Access Enable for Master TAG ID n]</b> [This bit enables a write access to the EMEM module region i SRAM addresses for transactions with the Master TAG ID n.] [[[0 <sub>B</sub> ] ][Write access will not be executed. ]] [[[1 <sub>B</sub> ] ][Write access will be executed. ]]

## Extension Memory (EMEM)

### [[[EMEM Module Region i Write Access Enable Register 1]

[[RGNACCENWB<sub>i</sub>] (i=0-7)]

[EMEM Module Region i Write Access Enable Register 1(0005C<sub>H</sub>+i\*10<sub>H</sub>)] [Application Reset] Value: FFFF FFFF<sub>H</sub>]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw															

[Field]	[Bits]	[Type]	[Description]
[[EN <sub>n</sub> ] (n=32-63)]	[n-32]	[rw]	<p><b>Access Enable for Master TAG ID n]</b>            [This bit enables a write access to the EMEM module region i SRAM addresses for transactions with the Master TAG ID n.]            [[[0<sub>B</sub>]]][Write access will not be executed. ]]            [[[1<sub>B</sub>]]][Write access will be executed. ]]</p>

### [[[EMEM Module Region i Read Access Enable Register 0]

[[RGNACCENRA<sub>i</sub>] (i=0-7)]

[EMEM Module Region i Read Access Enable Register 0(000D8<sub>H</sub>+i\*10<sub>H</sub>)] [Application Reset] Value: FFFF FFFF<sub>H</sub>]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	ENO
rw															

[Field]	[Bits]	[Type]	[Description]
[[EN <sub>n</sub> ] (n=0-31)]	[n]	[rw]	<p><b>Access Enable for Master TAG ID n]</b>            [This bit enables a read access to the EMEM module region i SRAM addresses for transactions with the Master TAG ID n.]            [[[0<sub>B</sub>]]][Read access will not be executed. ]]            [[[1<sub>B</sub>]]][Read access will be executed. ]]</p>

## Extension Memory (EMEM)

### [[[EMEM Module Region i Read Access Enable Register 1]]]

[[RGNACCENRB<sub>i</sub>] (i=0-7)]

[EMEM Module Region i Read Access Enable Register 1(000DC<sub>H</sub>+i\*10<sub>H</sub>)] [Application Reset] Value: FFFF  
FFFF<sub>H</sub>]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw															

.....

[Field]	[Bits]	[Type]	[Description]
[[EN <sub>n</sub> ] (n=32-63)]	[n-32]	[rw]	<p><b>[Access Enable for Master TAG ID n]</b></p> <p>[This bit enables a read access to the EMEM module region i SRAM addresses for transactions with the Master TAG ID n.]</p> <p>[[[0<sub>B</sub> ]][Read access will not be executed. ]]</p> <p>[[[1<sub>B</sub> ]][Read access will be executed. ]]</p>

---

**Extension Memory (EMEM)**

### 23.4.3 [[EMEM Module RAM]]

[[The [EMEM Module] RAM shall be accessed via the EMEM module SRI slave interface and BBB slave interface.]]

**Table 1134 [Register Address Space - EMEM\_RAM[ ]]**

[Module]	[Base Address]	[End Address]	[Note]
(fs0)	[00000000 <sub>H</sub> ]	[000FFFFF <sub>H</sub> ]	BBB slave interface 0 (access to EMEM module RAM, non-cached segment)
	[00000000 <sub>H</sub> ]	[000FFFFF <sub>H</sub> ]	BBB slave interface 0 (access to EMEM module RAM, cached segment)
(s0)	[00000000 <sub>H</sub> ]	[000FFFFF <sub>H</sub> ]	SRI slave interface 0 (access to EMEM module RAM, non-cached segment)
	[00000000 <sub>H</sub> ]	[000FFFFF <sub>H</sub> ]	SRI slave interface 0 (access to EMEM module RAM, cached segment)

---

**Extension Memory (EMEM)****23.4.4 [[EMEM XTM RAM]]**

[[The XTM RAM shall be accessed via the BBB fx slave interface.]

■■■

**Table 1135 [Register Address Space - XTM[ ]]**

[Module]	[Base Address]	[End Address]	[Note]
(fx)	[00000000 <sub>H</sub> ]	[0000FFFFE <sub>H</sub> ]	XTM FPI slave interface

## Extension Memory (EMEM)

## 23.5 [[Revision History]]

**Table 1136 [Changes]**

[Reference]	[Change to Previous Version]	[Comment]
<b>[V1.3.11]</b>		
[[ <a href="#">Page 12</a> ]]	[English grammar (use of 'a' & 'an').]	
[[ <a href="#">Page 15</a> ]]		
[[ <a href="#">Page 15</a> ]]	[English grammar (error is a noun).]	
<b>[V1.3.12]</b>		
[[ <a href="#">Page 36</a> ]]	[Revision History update.]	
<b>[V1.3.13]</b>		
[[ <a href="#">Page 12</a> ]]	[EMEM alarms.]	
[[ <a href="#">Page 9</a> ]]	[EMEM initialisation.]	
[[ <a href="#">Page 3</a> ]]	[Change device name.]	
<b>[V1.3.14]</b>		
[]	[No changes.]	[]
<b>[V1.4.1]</b>		
[[ <a href="#">Page 1</a> ]]	[Block diagram changed to include TC3Ax additions]	
[[ <a href="#">Page 3</a> ]]	[Add TC3Ax column and add rows for new interfaces]	
[[ <a href="#">Page 16</a> ]]	[Add SEP accesses for TC3Ax]	
[[ <a href="#">Page 7</a> ]]	[Remove constraint on order of TCM and XCM modules]	
[[ <a href="#">Page 20</a> ]]	[Add extra TILESTATE1 register to support increased memory size]	
[[ <a href="#">Page 20</a> ]]	[Add extra TILECONFIG1 register to support increased memory size]	
[[ <a href="#">Page 20</a> ]]	[In register TILECC bit field 23 to 16 changed from Reserved to XCMx (x=8-15).]	
[[ <a href="#">Page 15</a> ]]	[Typo corrected (missing blank).]	
<b>[V1.4.2]</b>		
[[ <a href="#">Page 1</a> ]]	[Block diagram changed to remove m0 SEP instance]	
[[ <a href="#">Page 3</a> ]]	[Removed sep_m0_* from configurations table]	
[[ <a href="#">Page 16</a> ]]	[Removed sep_m0_* from priority list of SEP interfaces. Added words "in this case" to clarify the scope of the priority list.]	
<b>[V1.4.3]</b>		
[[ <a href="#">Page 18</a> ]]	[Remove Registers TILESTATE1 and TILECONFIG1 from Chapter [ <a href="#">EMEM Core Register Description</a> ]]	

[]]]]

## Radar Interface (RIF)

### 24 Radar Interface (RIF)

RIF module connects one fast external ADC with up to four channels, or up to 4 single channel internal ADCs with an SPU (Signal Processing Unit) module. Each multi-channel external ADC is connected over LVDS pads conforming to IEEE 1596.3 General Purpose Link standard.

The group of four data signals shares one clock and one frame signal. The four channel data flow path consists of an ADC, quad deserializer performing serial to parallel conversion, quad FIFO and lane management block, and data memory interface.

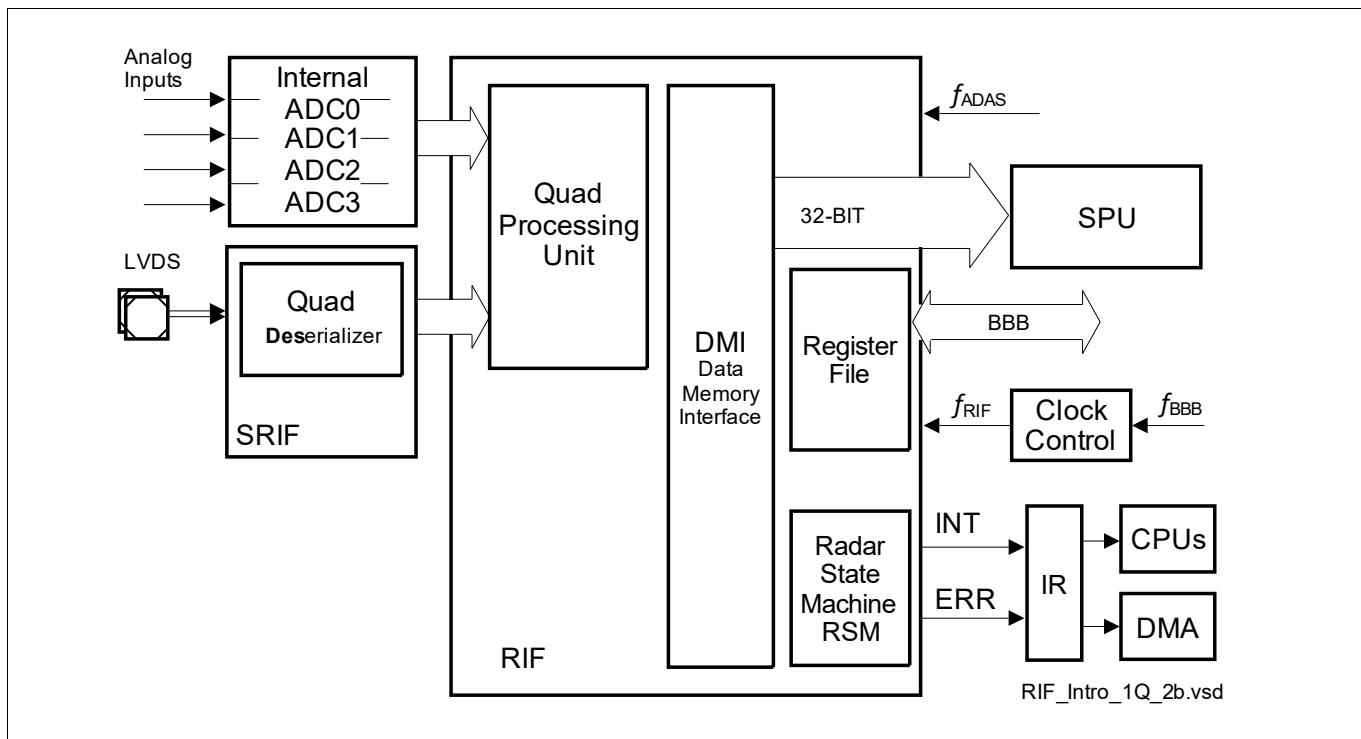


Figure 287 RIF Overview

#### 24.1 Feature List

This section describes the features of the RIF module.

- One interface for external four channel ADC
  - LVDS physical layer conform to IEEE-1596.3 standard, General Purpose Link
  - one frame signal, one clock signal, four data signals
  - independent frequency domains for each interface
  - independent deserializer unit
  - DDR (Double Data Rate) signalling with baud rate of up to 200MHz (400MBaud)
  - TC3Ax supports DDR signalling with baud rate of up to 300MHz (600MBaud)
  - data 10, 12, 14 or 16 bits wide, 2's complement format.
  - shift direction MSB or LSB first
  - data alignment left or right (integer or fractional)
- Interfaces to four internal single channel ADCs
  - 12-bit samples
- Intermediate layer providing data ordering and de-interleaving
  - Multiple lane data management

## Radar Interface (RIF)

- Real and complex sampling support
- De-interleaving
- Parallel memory interface to an internal SPU unit with FFT engine
  - 32-bit bus width
- Radar State Machine
  - 1, 2, 3 ... up to 2048 samples in a ramp
  - 1, 2, 3, ... up to 2048 ramps in a chirp
- Interrupts
  - Start of Ramp, End of RampCRC Error per channel

### 24.2 Overview

The RIF module chapter is based on describing the interfaces that connect the RIF module to its external and internal counterparts.

There are two data input interfaces (see [Figure 288](#)):

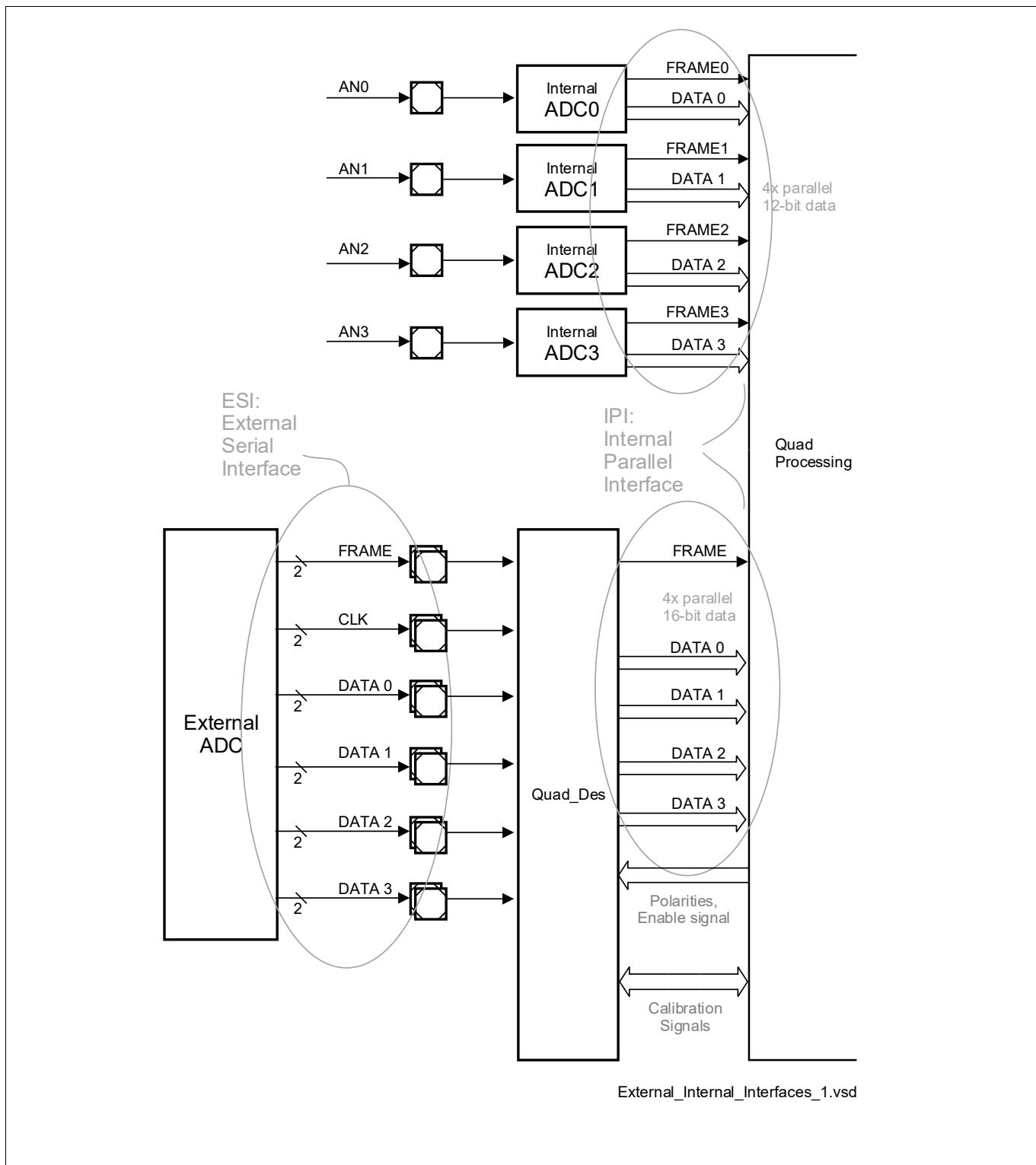
- External Serial Interface (ESI)
  - connects external ADCs to the quad deserializers
- Internal Parallel Interface (IPI)
  - connects the internal on-chip ADCs to the digital kernel of the RIF module
  - connects the quad deserializers to the digital kernel of the RIF module
- Data Memory Interface (DMI), see [Figure 305](#)

Additionally, the RIF module provides a Radar State Machine (RSM) monitoring the radar cycle.

**Table 1137 RIF Module Identification Numbers**

RIF Derivative	Module ID
RIF_TC33x	00E3 C001H
RIF_TC35x	00E3 C001H
RIF_TC39x	00E3 C001H
RIF_TC3Ax	00E3 C002H

### Radar Interface (RIF)



**Figure 288 Input Interfaces of the RIF Module**

## Radar Interface (RIF)

### 24.3 Functional Description

#### 24.3.1 External Serial Interface (ESI)

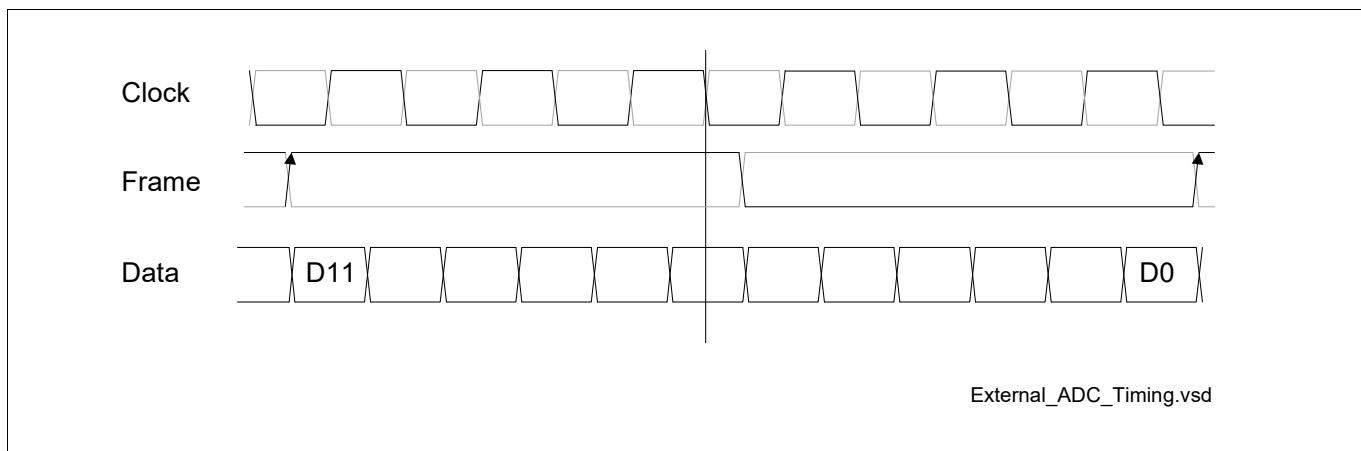
The external ADC interface consist of the following differential signals:

- Serial clock
- Frame
- Four data signals

The ADC provides continuous clock and continuous stream of data, back-to-back without pauses between the samples. This stream can only be made faster or slower by changing the sampling frequency delivered to the ADC.

The [Figure 289](#) shows an example of serial transmission of one 12-bit sample. The external ADC shifts the data bits approximately one half shift clock period later than the clock signal, in order to ensure sampling in the middle of the data bit time.

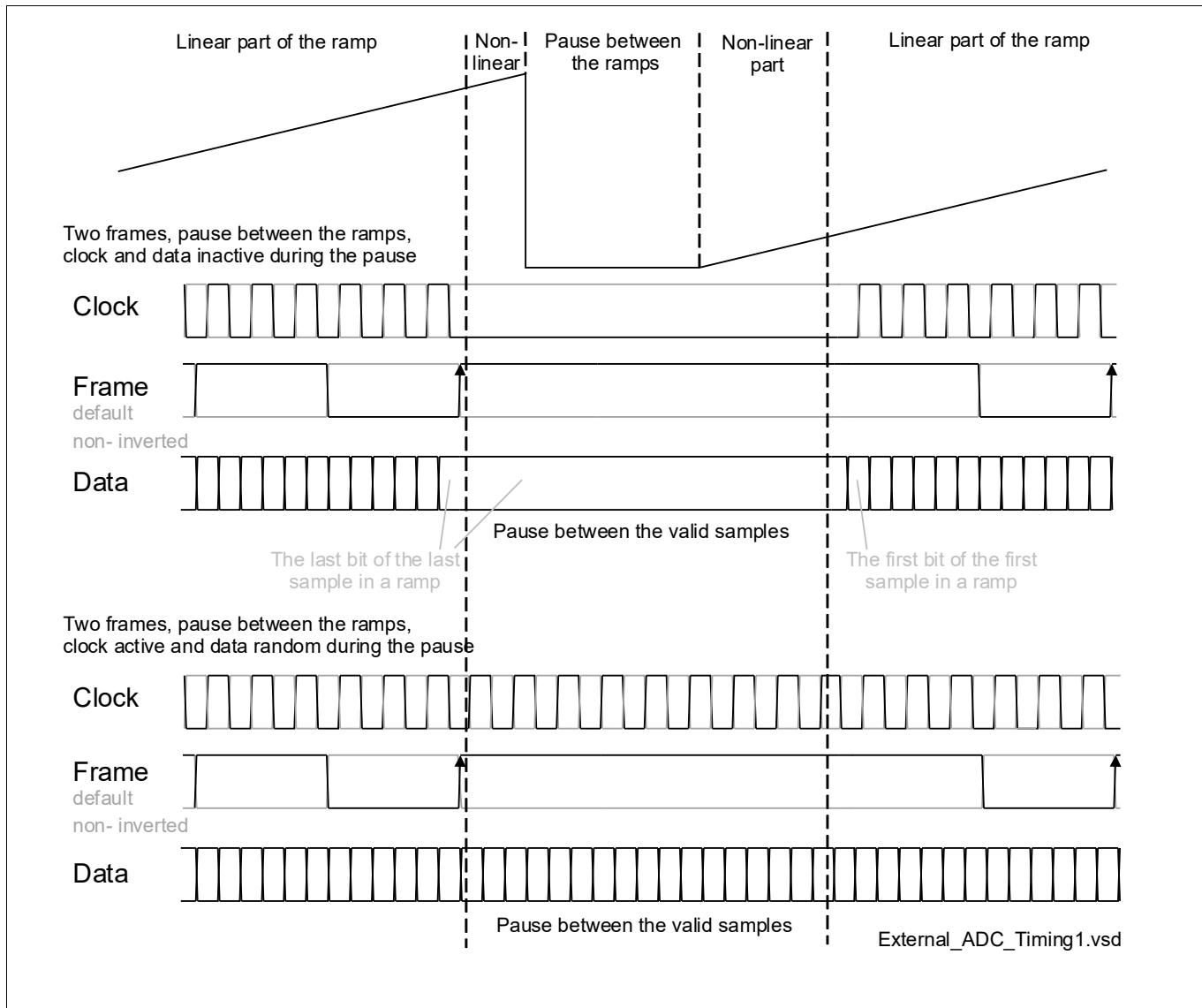
The frame signal marks each end of sample, per default with a rising edge (low-to-high differential transition). The frame signal toggles back in the middle of the sample, but this second edge does not have any special meaning.



**Figure 289 Waveforms of the External Serial Interface, Detailed View**

The [Figure 290](#) shows an overview of the waveforms in case a pause in the FRAME signal is used to mark the time interval between the ramps. Two scenarios are shown: one with the clock and data signals remaining static during the time between the linear/valid parts of the ramps, and one with the clock and data signals active.

## Radar Interface (RIF)



**Figure 290 Waveforms of the External Serial Interface, Overview**

### 24.3.2 Internal Parallel Interface (IPI)

The IPI interface consists of the following signals:

- Frame signal, used to write each sample in the FIFOs of the RIF kernel, see [Figure 288](#) and [Figure 293](#) for more detail.
- Data signals
  - 12-bit wide parallel data connection per internal ADC or
  - 16-bit wide parallel data connection per deserializer
- Configuration signals:
  - polarity of the clock, frame, and data
  - enable / disable = start / stop

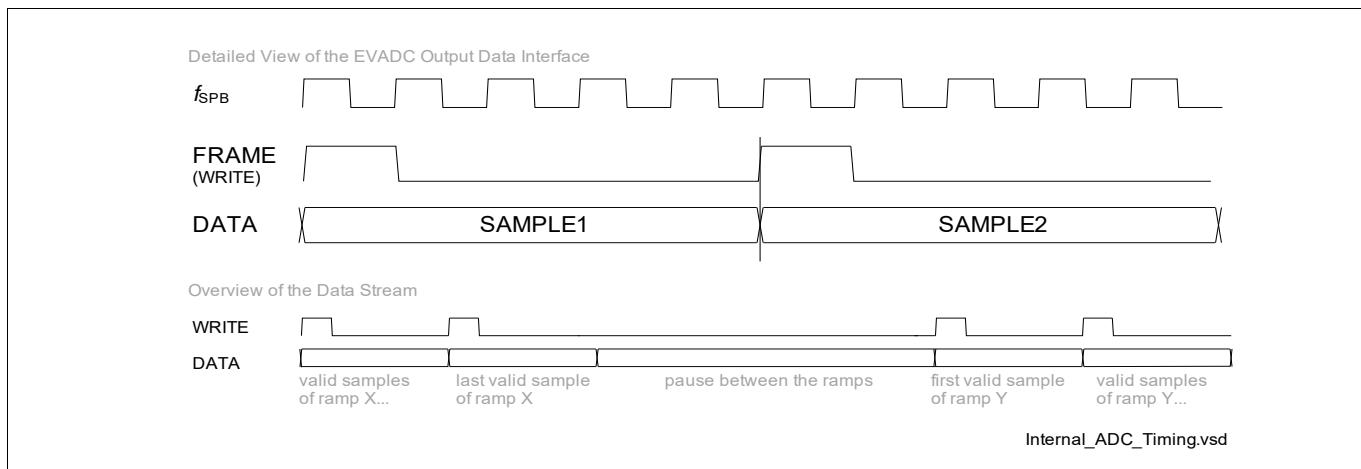
#### Connection to the Internal ADCs

The internal EVADCs (Enhanced Versatile ADCs) are single ended converters providing 12-bit unsigned integer data. Four internal EVADCs provide parallel data connection and FRAME (write) signal each.

## Radar Interface (RIF)

The **Figure 291** shows an overview of the waveforms generated by the EVADCs for writing their data into the RIF module. The FRAME (write) signal of the internal EVADC modules is  $1*f_{SPB}$  wide. Therefore, the internal RIF clock  $f_{ADAS}$  must be higher than  $f_{SPB}$ . For more details, see the section “Hardware Data Interface” in the EVADC module chapter.

The EVADCs provide two characteristic maximal sample rates, depending on the used clock frequency, of 2.6 MSamples/sec and 2.2 MSamples/sec. For more details of supported sampling rates, see the section “Conversion Timing Configurations” in the EVADC module chapter.



**Figure 291 Waveforms of the Internal Parallel Interface to the internal ADC**

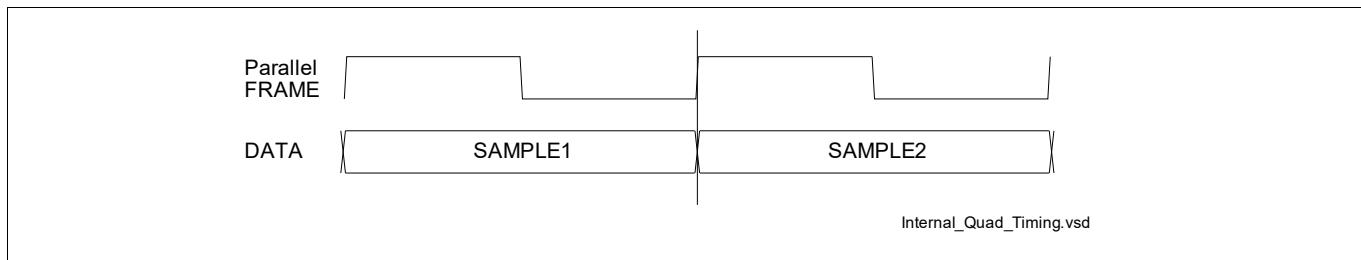
## Connection to the Quad Deserializer Unit

The Quad Deserializer Unit consists of four deserializers. A deserializer receives serial ADC data and delivers it in parallel form to the Quad Processing Unit.

The deserializer always delivers valid, non-corrupted samples. Dynamic start / stop during run time is allowed. After ending the stop state, the deserializer waits for the next frame and delivers valid samples.

Note that when the deserializer is disabled, the corresponding frame clock will be gated to low, which could trigger the Frame Watchdog to start and result in unintended Ramp1 error. Therefore for each radar cycle, in order to suppress the Ramp 1 error, after checking the R1EF, if there is no Ramp1 error it is recommended that the user will reset the Frame Watchdog threshold to 0 until the next radar cycle before disabling the deserializer.

The **Figure 292** shows an overview of the waveforms generated by the serializer for writing its received data into the RIF module by using the Internal Parallel Interface.



**Figure 292 Waveforms of the Internal Parallel Interface to the Quad Deserializer**

## Radar Interface (RIF)

### 24.3.3 Quad Processing Unit

After entering the frequency domain of the Quad, the data goes through a processing pipeline consisting of synchronization FIFOs, CRC check, FIFO and Lane Management block FLM and Data Formatting Unit DFU.

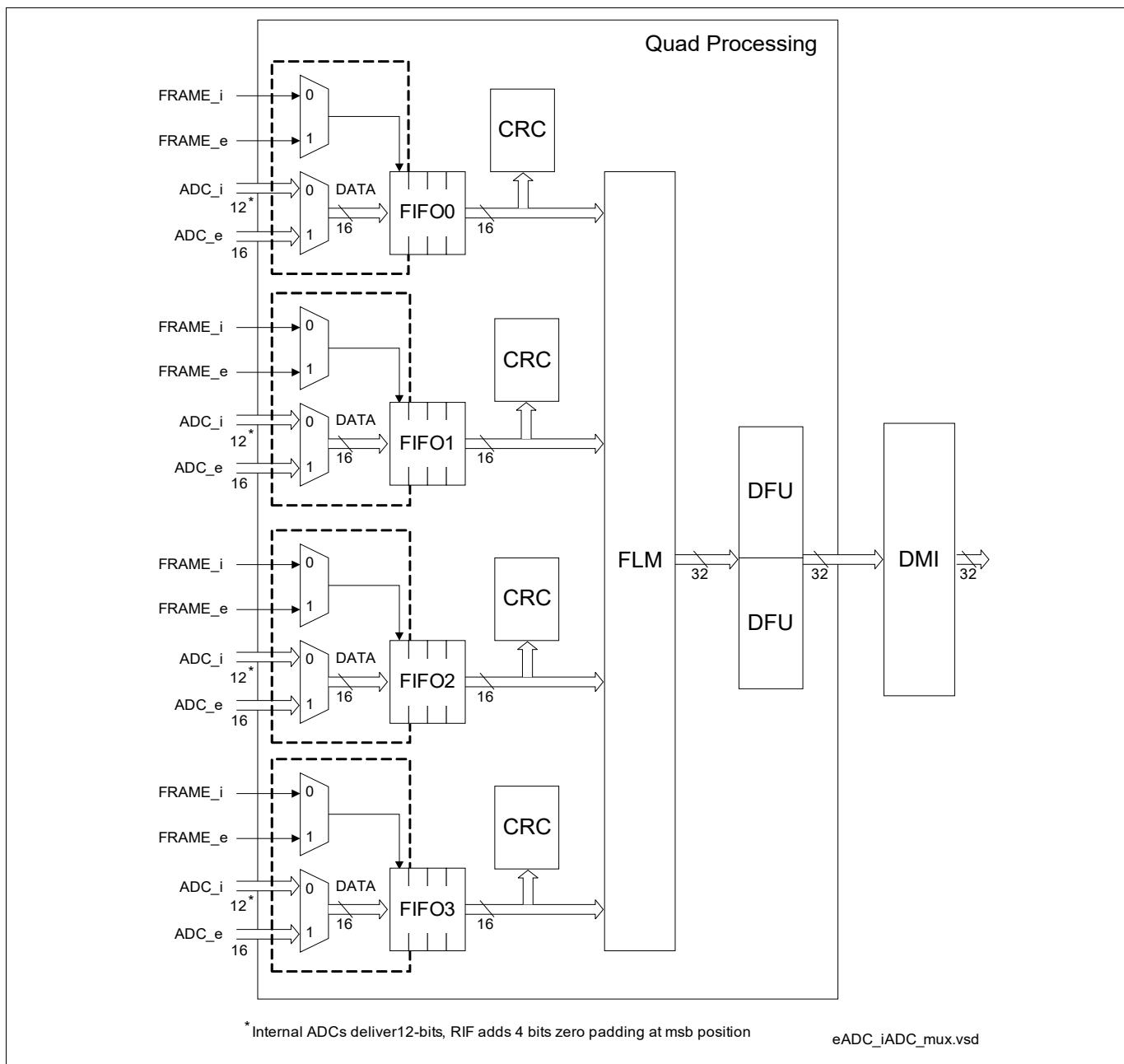


Figure 293 Quad Processing Unit

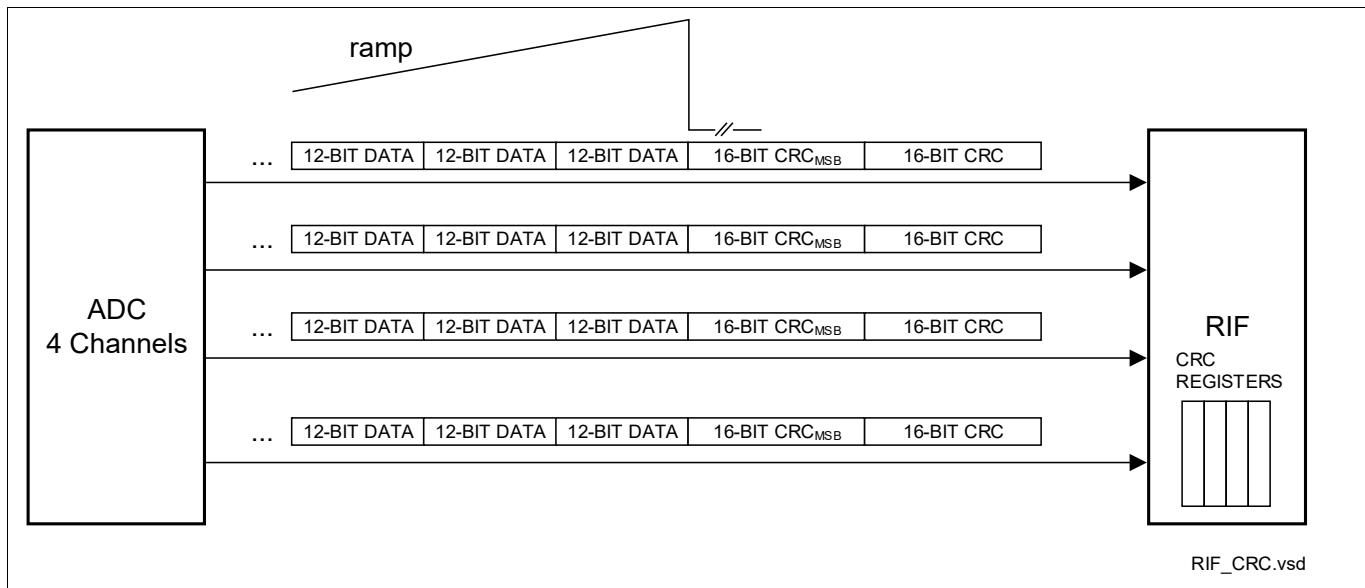
### 24.3.4 Default CRC Scheme

Each data channel contains one CRC (Cyclic Redundancy Check) calculation block which can be enabled or disabled. The CRC engine uses the 32-bit CRC polynomial 0x04C1 1DB7 with the initial value 0xFFFF FFFF. The external ADCs optionally provide the CRC in each data channel, at the end of the ramp. The CRC is transmitted MSB first. The CRC feature can be enabled or disabled by using the bit **FLM.CRCEN**.

If a CRC error occurs in a channel, an error interrupt is raised, if enabled by using the bit fields **INTCON.CRCE0 ... 3**. The CRC is transmitted in two 16-bit frames, because the size of the receive shift register is 16-bit. In case there is no external RAMP1 signal, the RIF module uses the FRAME watchdog and distinguishes between data and CRC

## Radar Interface (RIF)

samples by counting the predefined number of data samples and then taking the next two frames as the CRC for the ramp. Note that TC3Ax does not support Ramp1 signal and relies on the FRAME watchdog to distinguish between data and CRC samples. The order in which the two CRC frames are sent is configured with the bit **FLM.EXPCRCWO**.



**Figure 294 CRC Overview**

**Attention:** *The internal ADCs do not generate CRC. It is the application software responsibility not to enable the CRC feature when using the internal ADCs.*

## Radar Interface (RIF)

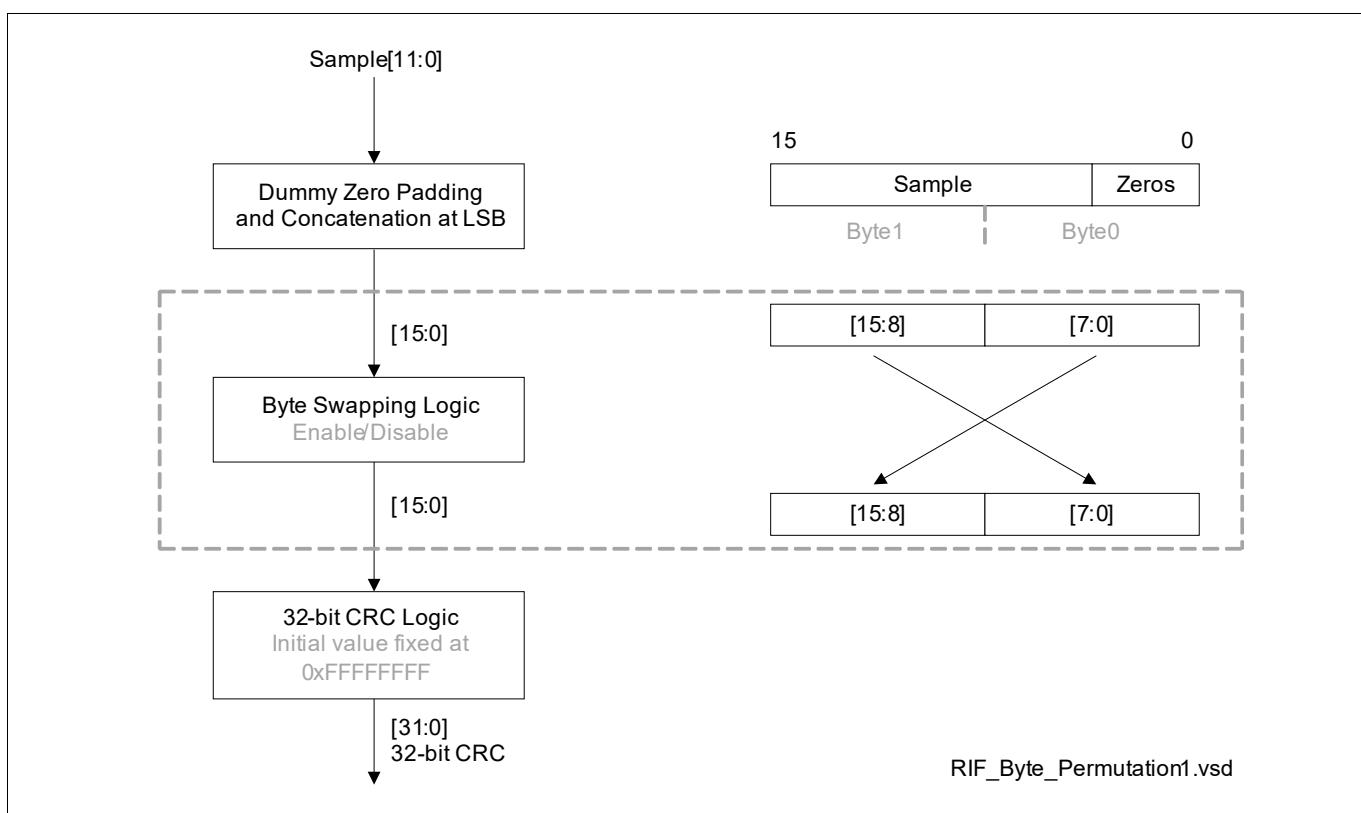
### 24.3.5 Alternative CRC Scheme

The RIF module provides an alternative way of calculating CRC. The features of this alternative CRC are:

- The CRC polynomial and the initial value are the same as the basic CRC. It is the 32-bit polynomial CRC 0x04C1 1DB7 with initial value 0xFFFF FFFF.
  - The RIF module pads the received ADC values with zeros on the LSB side to 16-bit and uses the padded values for calculating the CRC
  - The CRC is calculated always in the msb first direction independent of the shift direction on the serial bus
- The Alternative CRC Scheme can be selected by setting the bit **FLM.CRCALT**.

#### 24.3.5.1 Byte Swapping

If the bit **FLM.CRCALT=1**, then the bit **FLM.CRCSB** provides an option to swap the bytes of each sample before writing them to the CRC engine. The **Figure 295** shows an example of byte swapping of a 12-bit ADC sample, extended to 16-bit. The byte swapping operates independently of the sample width: 10, 12 , 14 or 16-bit.



**Figure 295 CRC Byte Swapping**

### 24.3.6 CRC as a Safety Mechanism

Additionally to covering EMI disturbances, the serial CRC can be used as a basis for software safety mechanism for checking the correct operation of the deserializer.

## Radar Interface (RIF)

### 24.3.7 Data Formatting Unit (DFU)

The following configuration parameters of DFU are programmable:

Input data properties:

- shift direction msb/lsb first
- even total data width of 10, 12, 14, 16 bits:
  - signed / unsigned integer of 10, 12, 14, 16 bits
  - unsigned Q1.9, Q1.11, Q1.13 and Q1.15 (total width 10, 12, 14, 16 bits)
  - signed Q1.8, Q1.10, Q1.12, Q1.14 (total width 10, 12, 14, 16 bits)
- Output data properties:
  - alignment: left / right (fractional / integer data)
  - data length fixed to 16-bit
  - data format fixed to signed integer

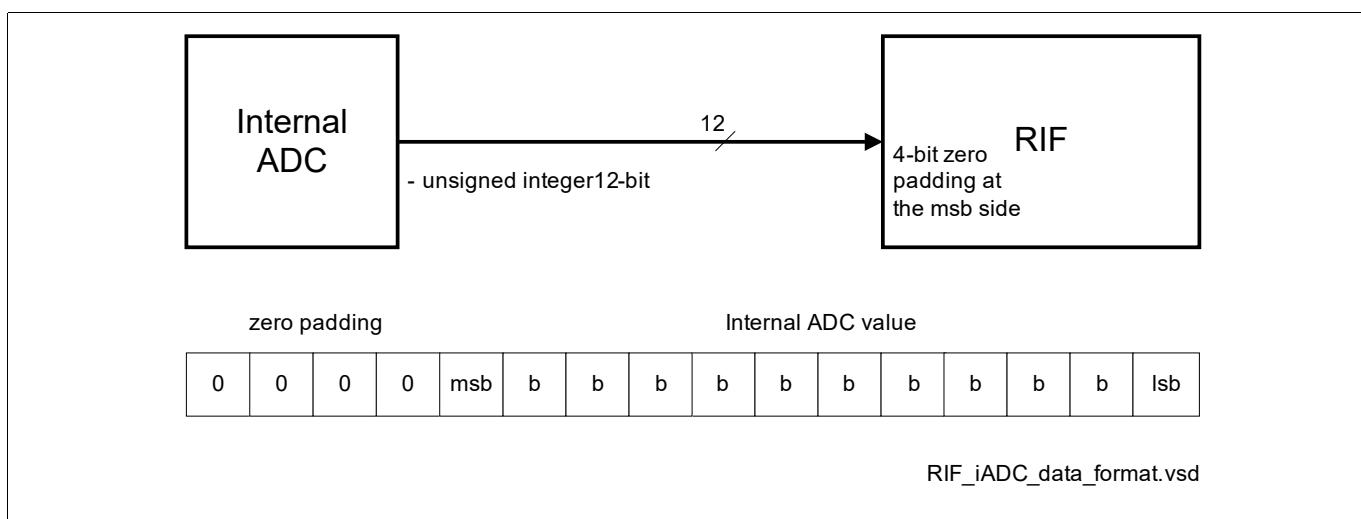
The parameters are initialized at the start of the application, after reset, and then they remain constant either until the end of the application (the power down or the next reset) or the module is stopped for configuration change and then restarted again. The frame length is the only parameter requiring dynamic reconfiguration.

There is one set of configuration bits valid for all four channels. Separate format of each channel is not possible.

Regarding the fixed point Q data formats, the data length of unsigned Qm.n format is m+n bits, and the data length of signed Qm.n format is m+n+1. This means that for example, Q1.11 datum is 12 bit long as unsigned, and 13 bit long as signed. Due to the fact that RIF can receive only even width data, signed Q1.11 is not supported, but Q1.10 or Q1.12 are supported, because they have even total data width.

This raw received data is always right aligned in its 16-bit slot, and in case of signed integer input data format, the RIF sets the padding msb bits, which are per default zeros, to the sign (the msb bit) of the raw input data.

The [Figure 296](#) shows the data format delivered by the internal ADC: 12-bit unsigned integer padded by RIF with zeros on the msb side. The reset values of the bits **DFU.DF** and **MSB** correspond to this data type.



**Figure 296 Data Format from the Internal ADC**

The [Figure 297](#) shows the overview of all data formats in the data path, starting with an external ADC.

## Radar Interface (RIF)

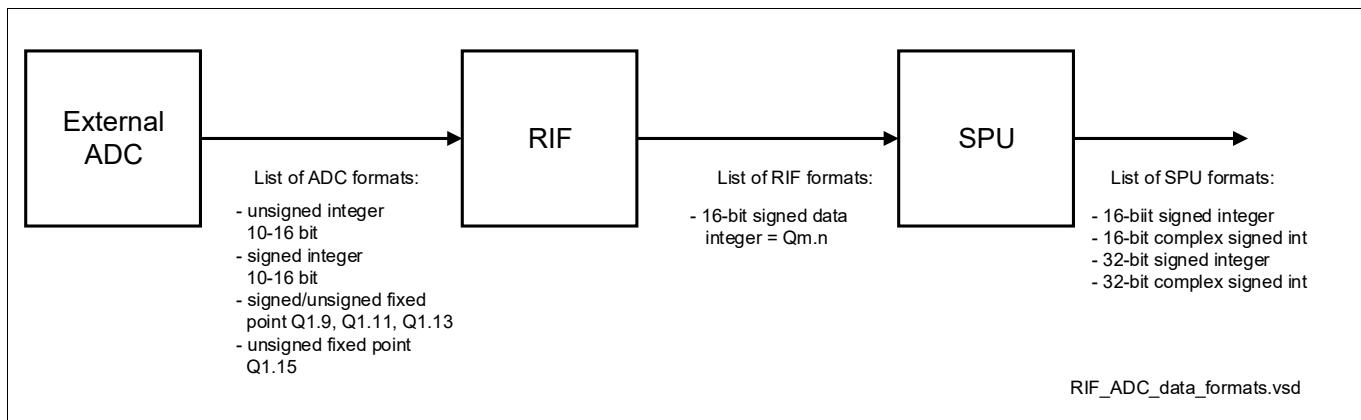


Figure 297 Data Formats from the External ADC

The RIF deserializer receives the data from the external ADC and delivers the data as shown in [Figure 298](#). If the serial data comes msb first, the lsb bit is on the parallel lsb position and no further operation is necessary. If the serial data comes lsb first, then the msb bit is on the parallel lsb position and the DFU has to bit reverse the data (see [DFU.MSB](#)).

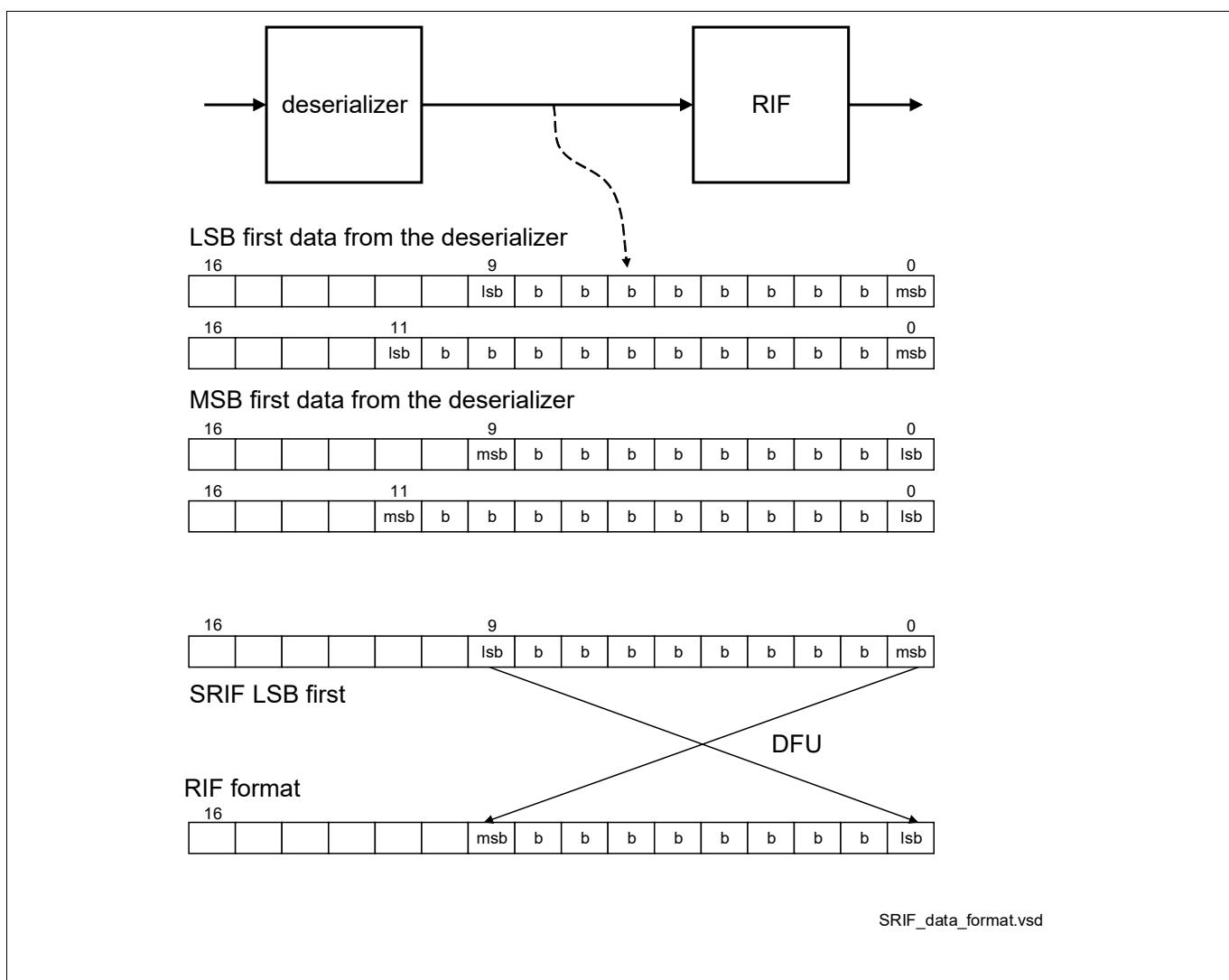


Figure 298 Shift Direction Management

## Radar Interface (RIF)

After optionally bit-reversing the data, the DFU can further re-format the data (see [DFU.DA](#)).

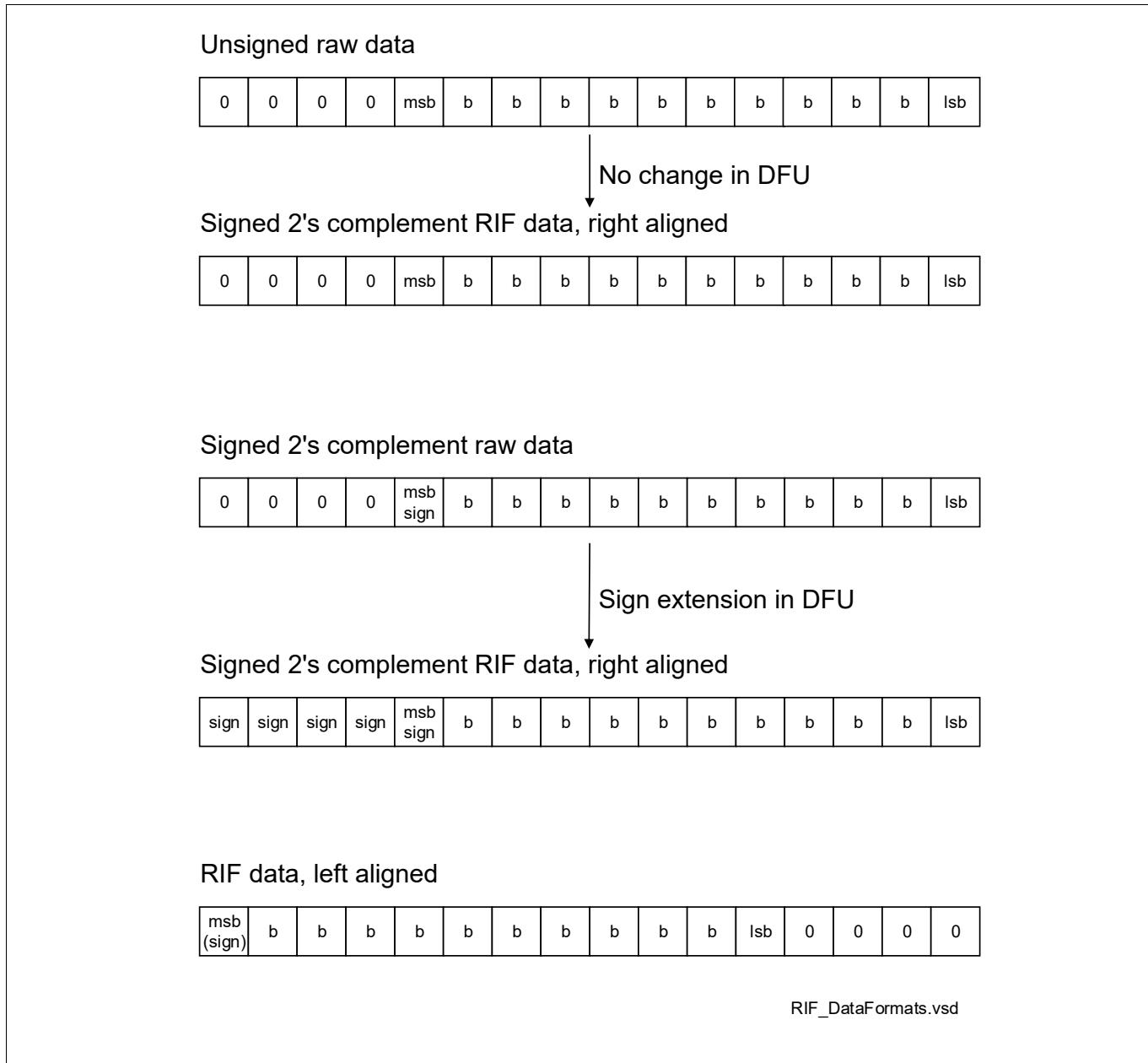


Figure 299 DFU Processing of the Data Formats

### 24.3.8 FIFO and Lane Management (FLM)

The streams of samples from external ADC can be transmitted over one, two or four lanes, depending on the sampling rate. Additionally, radar signals can be sampled either as real only, or in quadrature, resulting in real samples or complex samples (consisting of real and imaginary sample).

The samples of these streams are distributed in certain order in the Data Memory Interface, as described in the following subsections.

#### 24.3.8.1 FLM Operating Modes

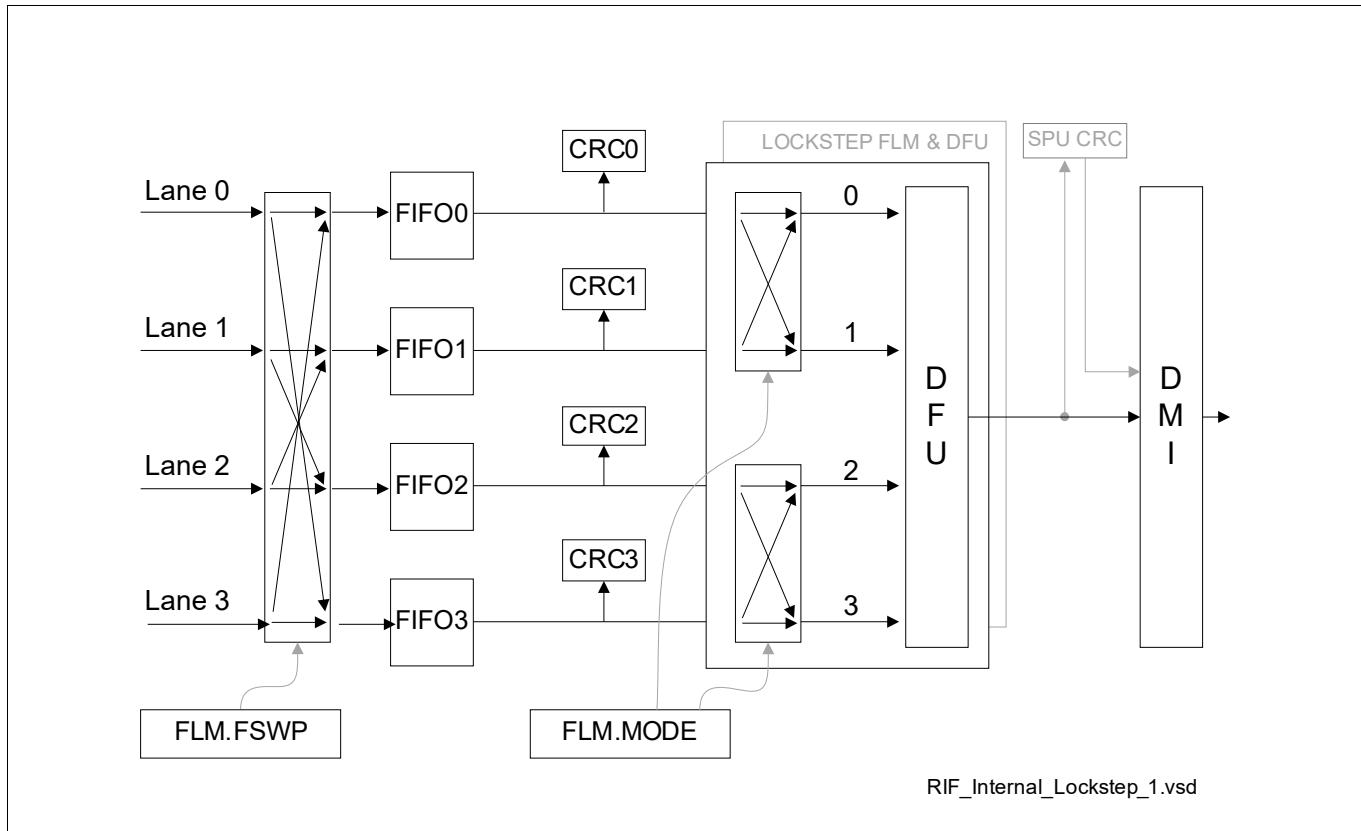
Each Quad\_FLM supports the following operating modes (see also register [FLM](#)):

- Direct complex mapping (radar front end chip soldered at the same side of the PCB with the uC)

## Radar Interface (RIF)

- Flipped complex mapping (radar front end chip soldered at the other side of the PCB).

The purpose of the flipping is to ensure always the same position of the real and complex part when delivered to the SPU.



**Figure 300 FLM Operating Modes**

The analog part of the quad, Quad\_Des, is not affected by the difference between the use cases of real and complex sampling. The different handling of the samples is done in the digital part of the quad, QUAD\_FLM.

In case of real sampling, if one antenna is being used, then it must be connected to lane 0. If two antennas are being used, they must be connected to lanes 0 and 1, for three antennas, lanes 0 to 2 must be used, for four antennas, lanes 0 to 4. Use [IPI.EN0 ... 3](#) accordingly.

In case of complex sampling, if one antenna is being used, it has to be connected to lanes 0 and 1. If a second antenna is being used, it has to be connected to lanes 2 and 3.

TC3Ax supports I and Q samples transmitted from the MMICs interleaved on a single LVDS lane. This feature can be enabled or disabled by [FLM.MULCMPX](#). When this feature is enabled, if one antenna is used, it has to be connected to lane 0. If two antennas are used, they must be connected to lanes 0 and 1, for three antennas, lanes 0 to 2 and for four antennas, lanes 0 to 3 must be used. A detailed description of this feature is available in [Section 24.3.8.5](#).

### 24.3.8.2 RIF Internal Lockstep and SPU CRC)

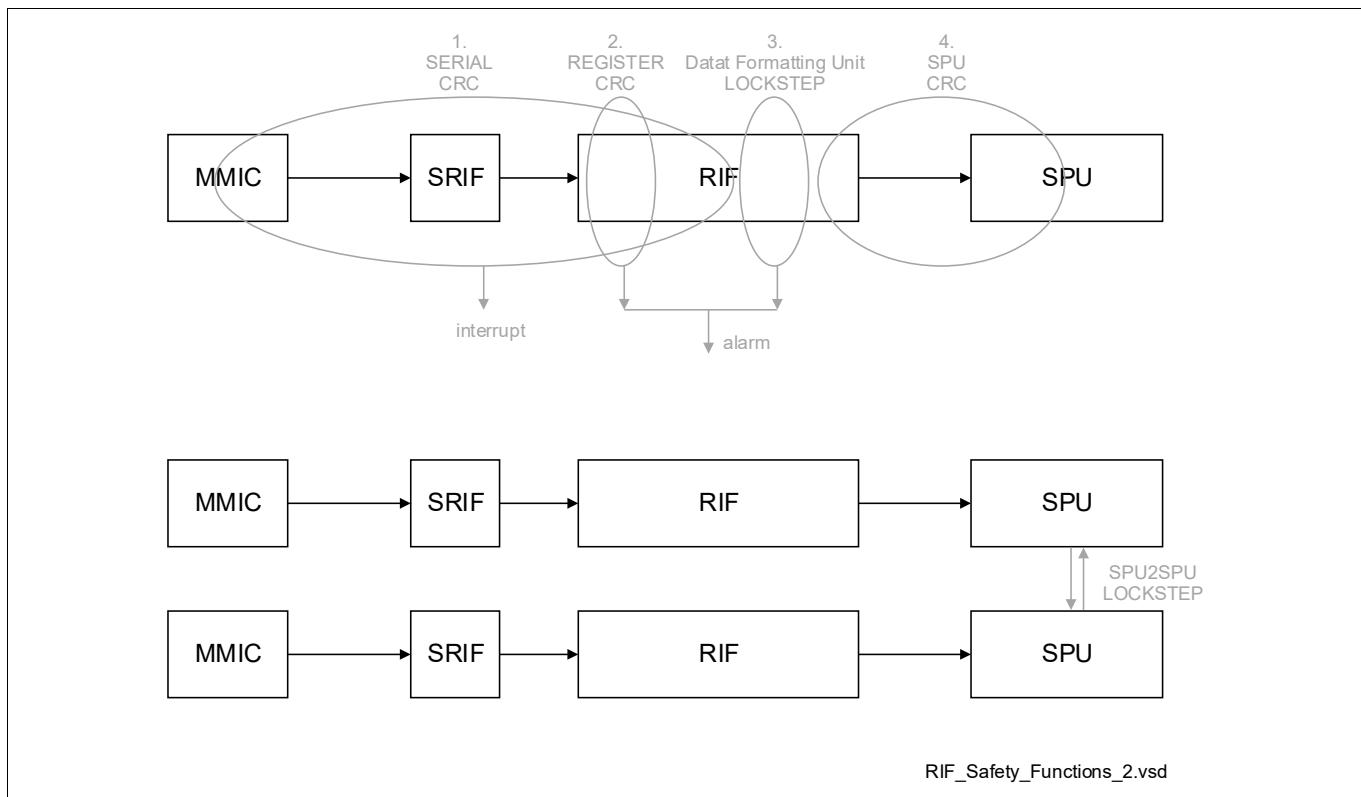
The RIF module provides an optional safety feature providing increased data path integrity. The safety feature consists of:

- a lockstep function covering the data multiplexers that is controlled by the bit [FLM.MODE](#) and the DFU unit
- 32-bit CRC is compatible with the TriCore CRC-32 implemented algorithm. It is done on the 32-bit RIF data output to the SPU.

## Radar Interface (RIF)

The safety feature is disabled by default, and can be enabled by software by using the bit-fields **SFCON.SPUCRC** and **LOCKI**. When enabled, the CRC values are attached at the end of the data stream for each ramp as a 32-bit value, and delivered in one move to the SPU. If the lockstep comparator detects mismatch, an SMU alarm is triggered.

The [Figure 301](#) shows an overview of all safety functions in the RIF module.



**Figure 301 Safety Functions Overview**

## Radar Interface (RIF)

## 24.3.8.3 Real and Complex Sampling

The diagram below describes two scenarios: first, where each of the four lanes delivers one real sample (a, b c, d), and second, where two lanes deliver one complex sample (r and i), and the second two lanes second complex sample (R and I).

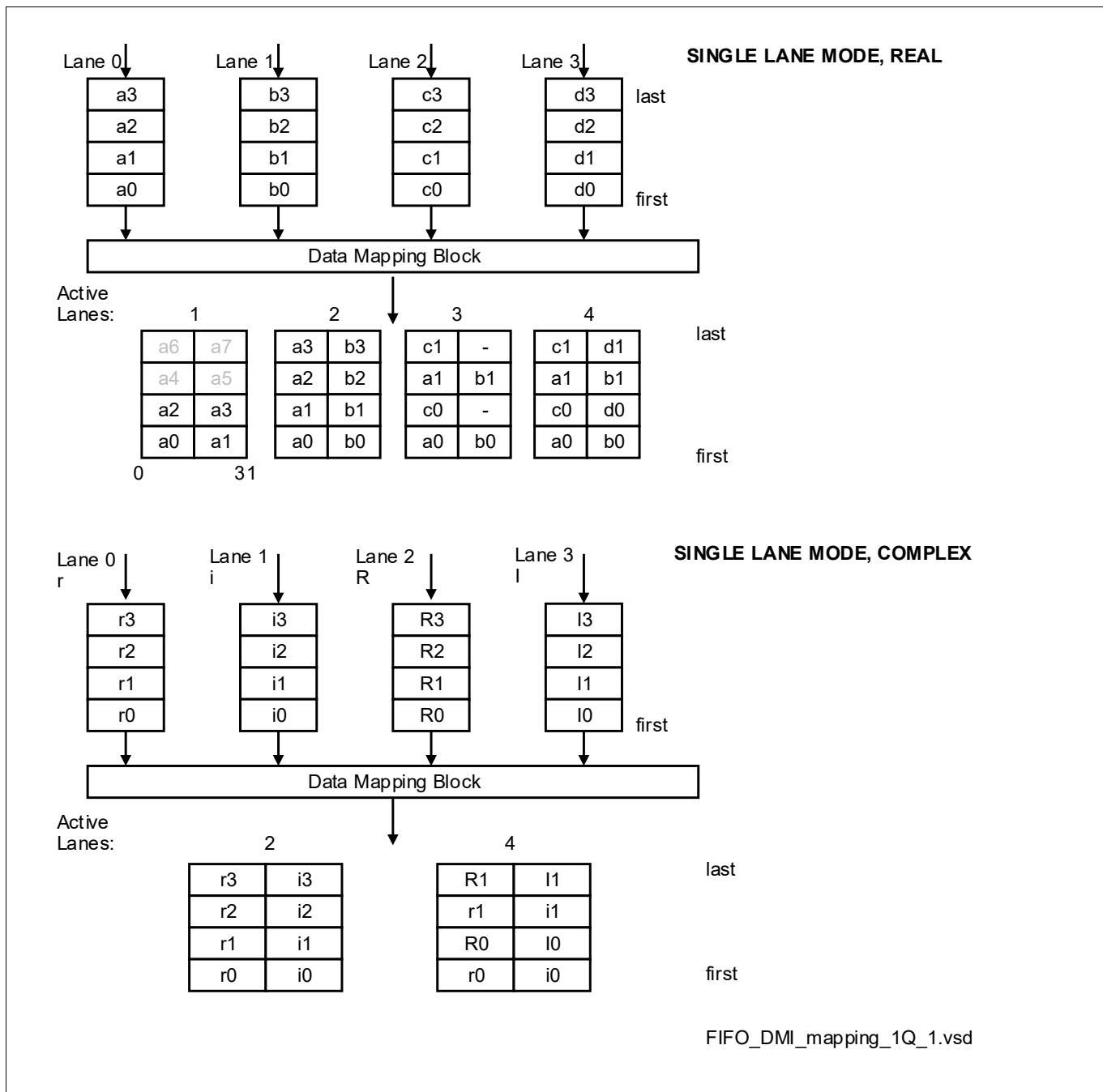


Figure 302 Single Lane, Real and Complex Sampling

## Radar Interface (RIF)

### 24.3.8.4 Multi Lane Real Sampling

The diagram below describes two scenarios: first, where pairs of FIFOs deliver high speed streams, and second, where four FIFOs deliver very high speed streams from one ADC.

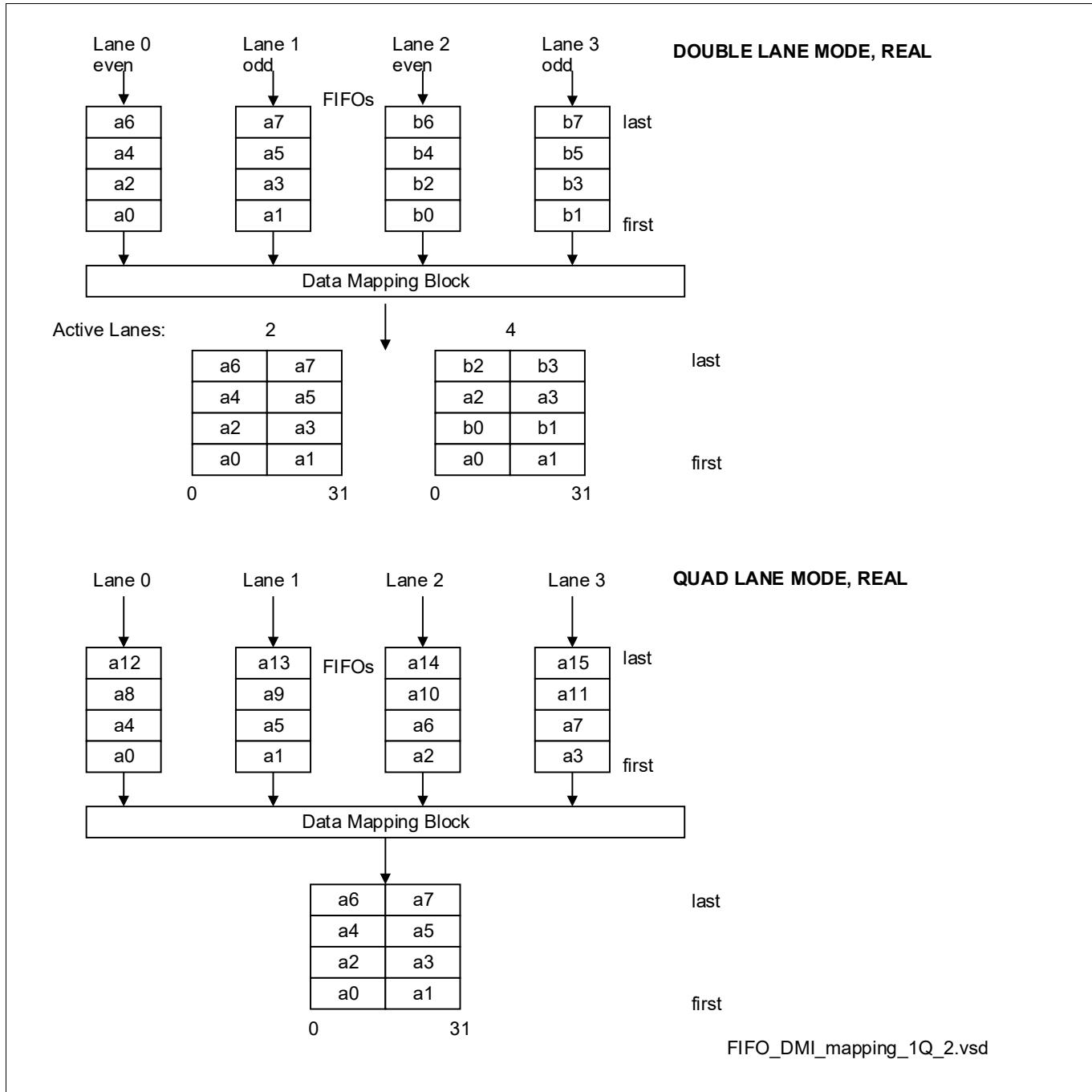


Figure 303 Multi Lane Real Sampling

### 24.3.8.5 Multiplex Complex Mode

This mode is only supported for TC3Ax. It can be activated by setting the **FLM.MULCMPX** bit to 1. In this mode, each of the four lanes deliver both the real and the imaginary components of the complex samples successively. The diagram below describes two scenarios: first, real component ( $r, R, ar, AR$ ) of the complex sample arrives first, followed by the imaginary component ( $i, I, ai, AI$ ), and the second where imaginary component of the complex sample arrives first, followed by the real component. This can be controlled using the bitfield **FLM.EXPIQ**. Note

## Radar Interface (RIF)

that **FLM.EXPIQ** can only be set to 1 when each LVDS lane is set deliver complex data by setting the bitfield **FLM.MULCMPX** to 1. Moreover, current internal ADCs are not capable of transmitting I and Q samples on a single LVDS lane. Therefore, use of external ADCs which are capable of interleaving I and Q data on a single LVDS lane is required to implement this mode.

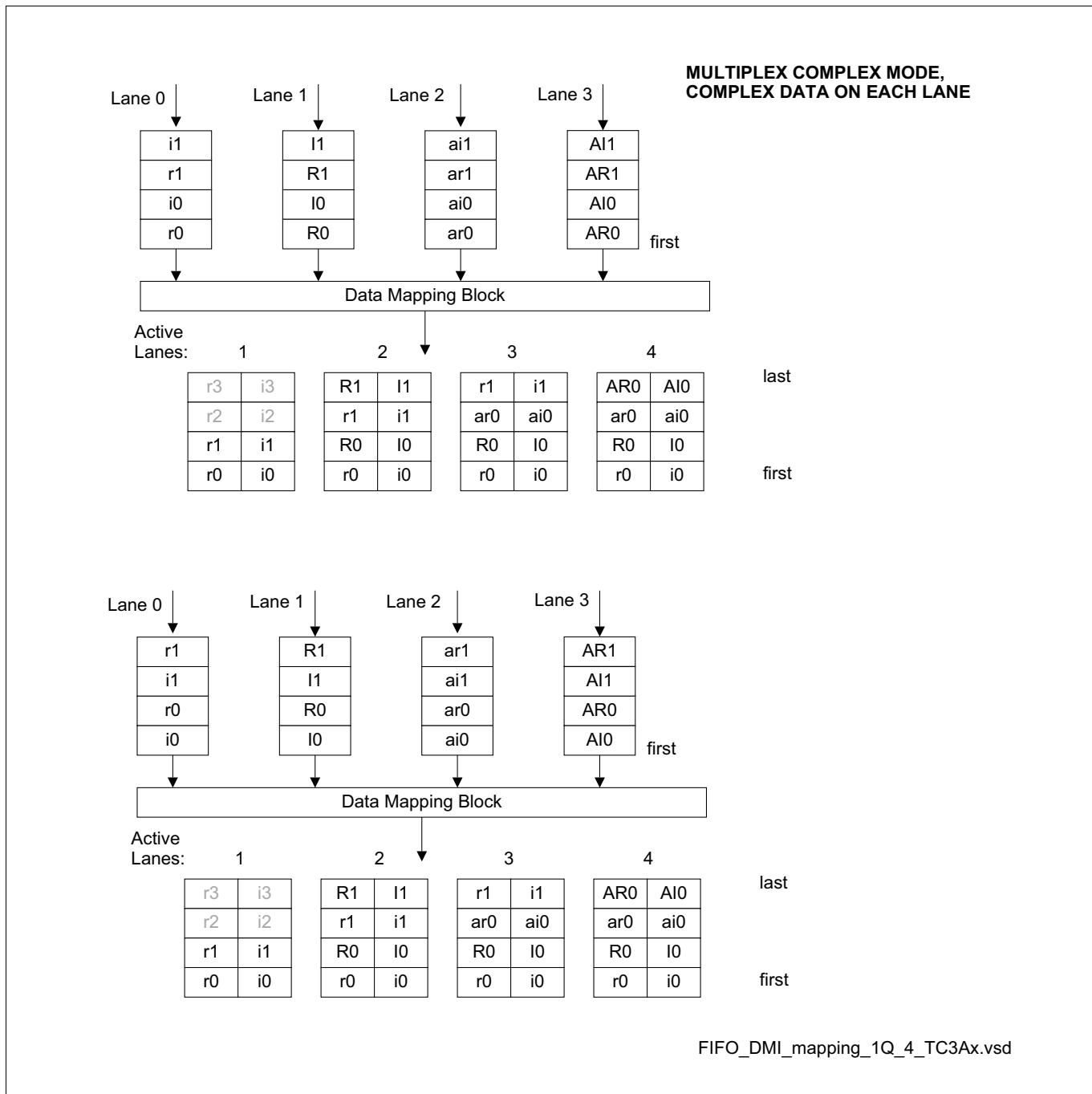


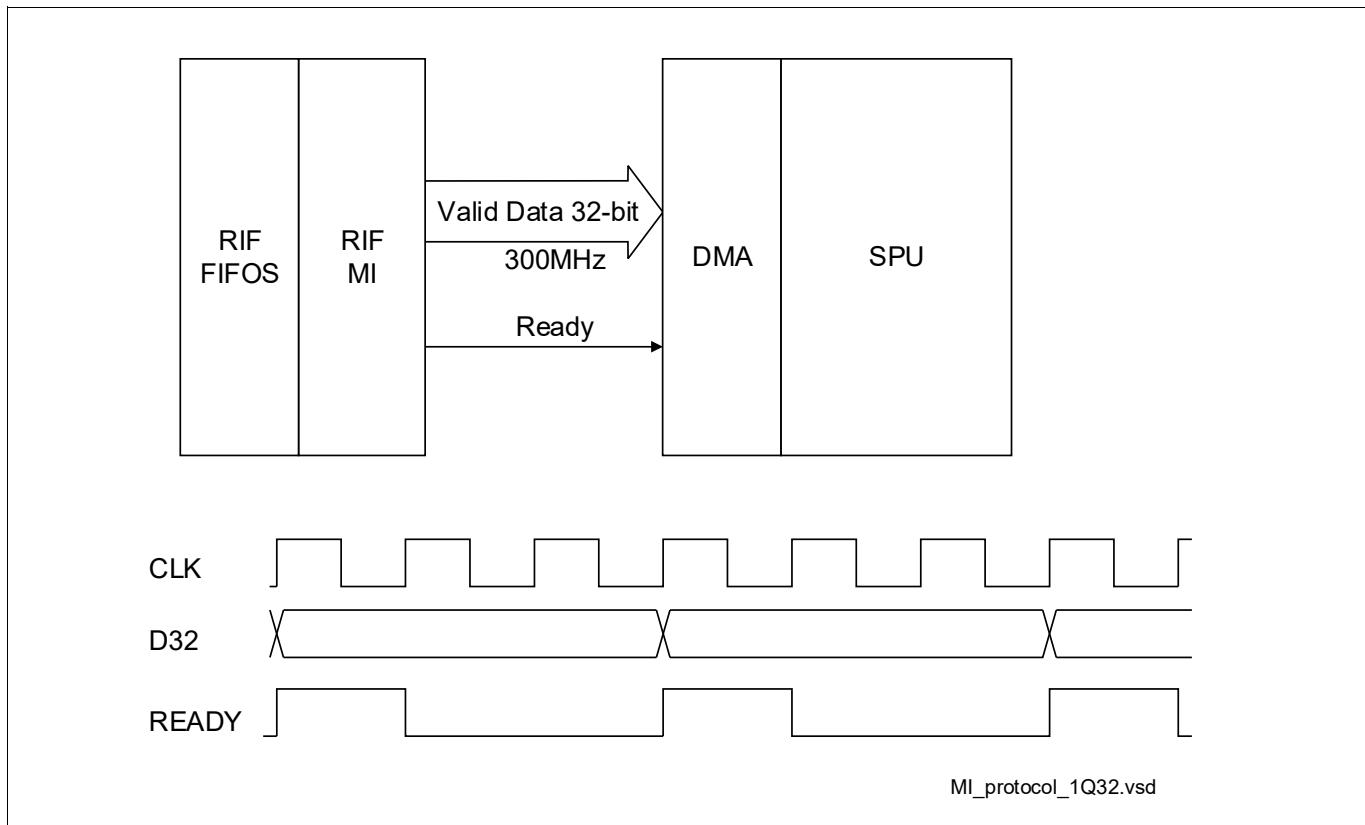
Figure 304 Multiplex Complex Sampling for Tc3Ax

## Radar Interface (RIF)

### 24.3.9 Data Memory Interface (DMI)

The data interface to the internal FFT provides the following signals:

- Output Signals
  - 32-bit data
  - READY signal



**Figure 305 Data Memory Interface (DMI)**

#### 24.3.9.1 Data Format of the Memory Interface

The ADC samples are ordered in Little Endian order in the memory interface, formatted as configured in the Data Formatting Unit.

## Radar Interface (RIF)

### 24.3.10 Radar State Machine (RSM)

The radar state machine RSM provides monitoring of the radar operation cycle.

The radar use-cases can be sub-divided into two classes: “[External ADC Use-Case](#)” and “[Internal ADCs Use-Case](#)”.

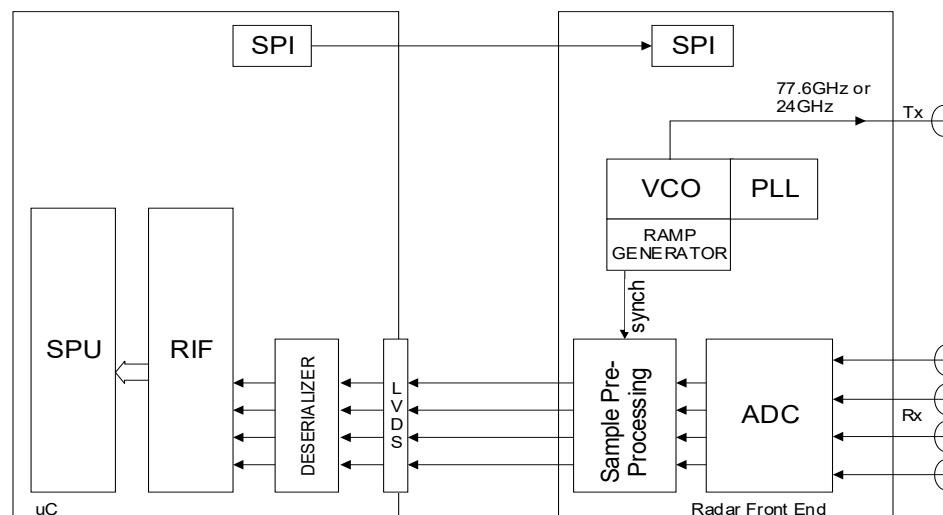
### 24.3.11 External ADC Use-Case

The optimal external system provides a pre-processed set of ADC samples for each ramp of the RF transmitter, filtered to remove invalid samples, and terminated with a CRC compatible with the CRC checker integrated into the RIF. The CRC is optional but allows the integrity of the data transferred into the RIF to be verified. In this kind of system, the RIF Frame Watchdog can be used to maintain or verify synchronisation of the RIF with the frequency ramps of the external RF components

For legacy systems using “free running” ADCs which sample continuously, the external system can provide the optional RAMP1 signal marking the start and the end of the linear part of each ramp. The RAMP1 signal is active only during the valid part of the frequency ramp and allows the RIF to filter the incoming stream of samples to ensure that only valid samples are. Interrupts can be triggered, if enabled.

The RAMP1 signal generated by the radar front-end must satisfy the sample and hold requirements relative to the FRAME signals as defined in the datasheet. Because this timing puts limit on the maximum achievable baud rate, and uses more pins, it is recommended to use the watchdog timer mode using the Frame Watchdog.

In TC3Ax, Ramp1 signal is not supported and Frame Watchdog must be used instead.



**Figure 306 System View: External ADCs with Pre-processing**

## Radar Interface (RIF)

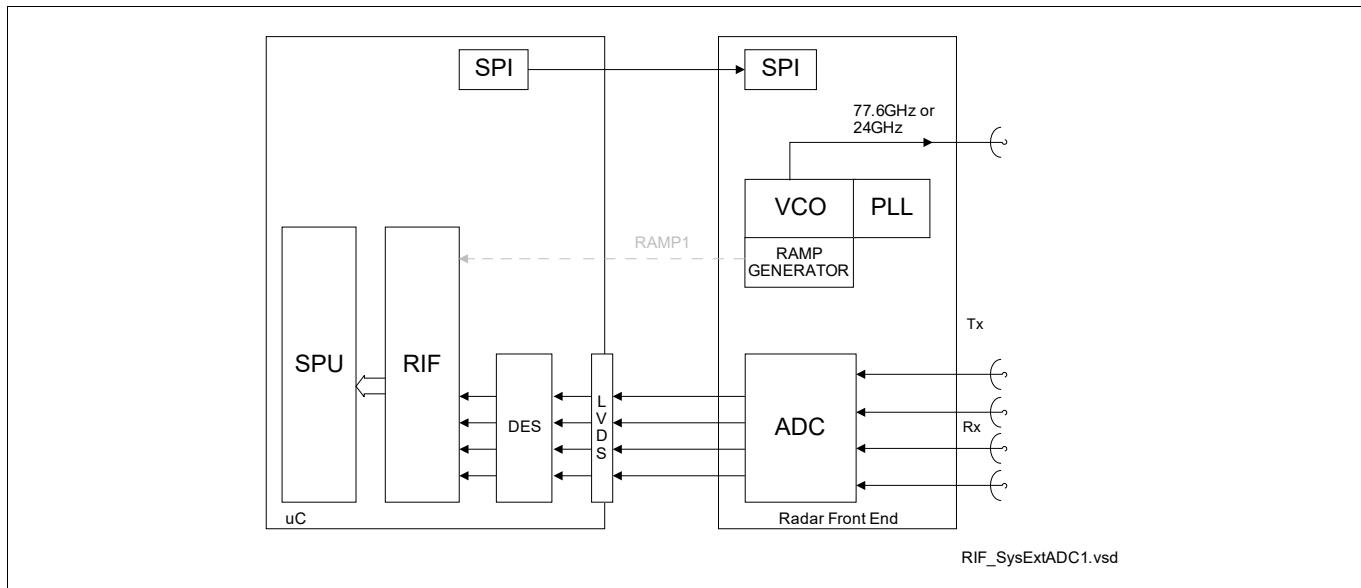


Figure 307 System View, External ADCs Use-Case

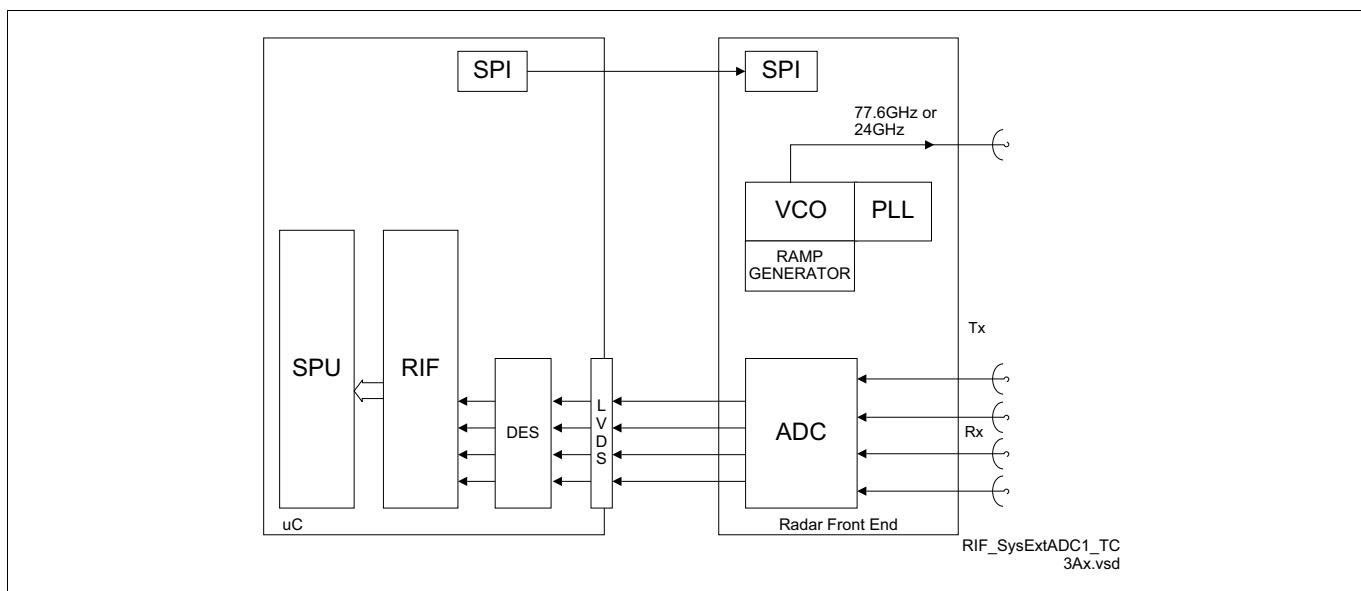
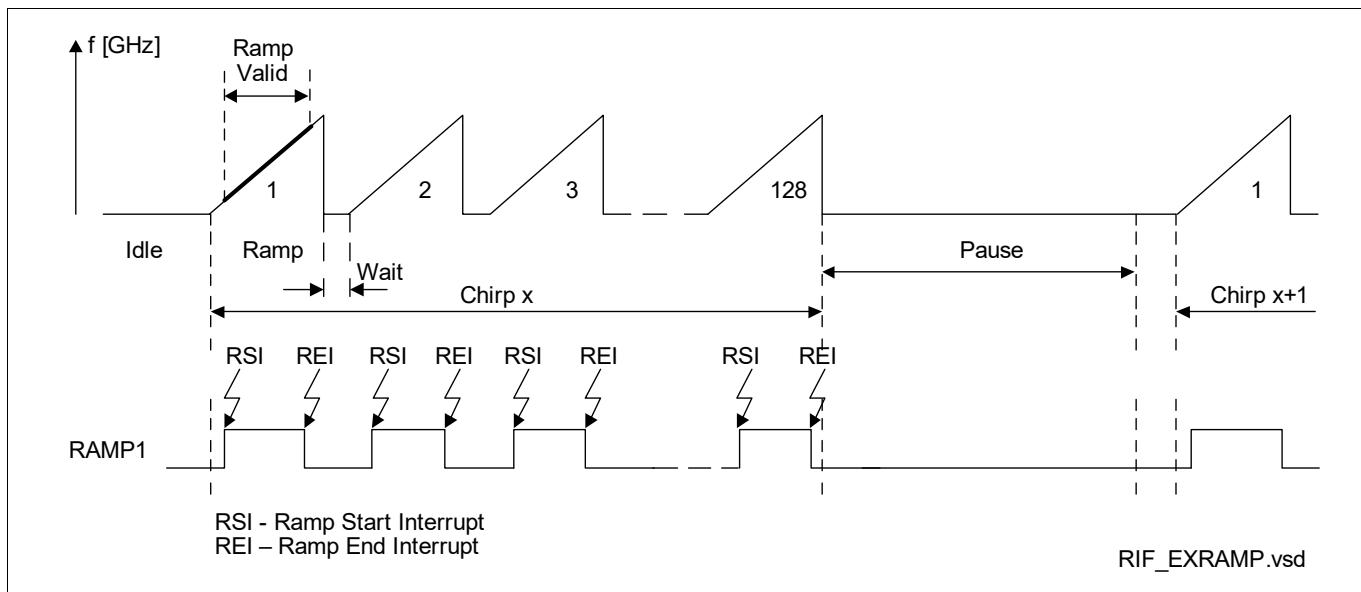


Figure 308 System View, TC3Ax External ADCs Use-Case

## Radar Interface (RIF)



**Figure 309 Radar Operating Cycle, External Radar Front-End**

The radar cycle is a simple sequence consisting of the following steps:

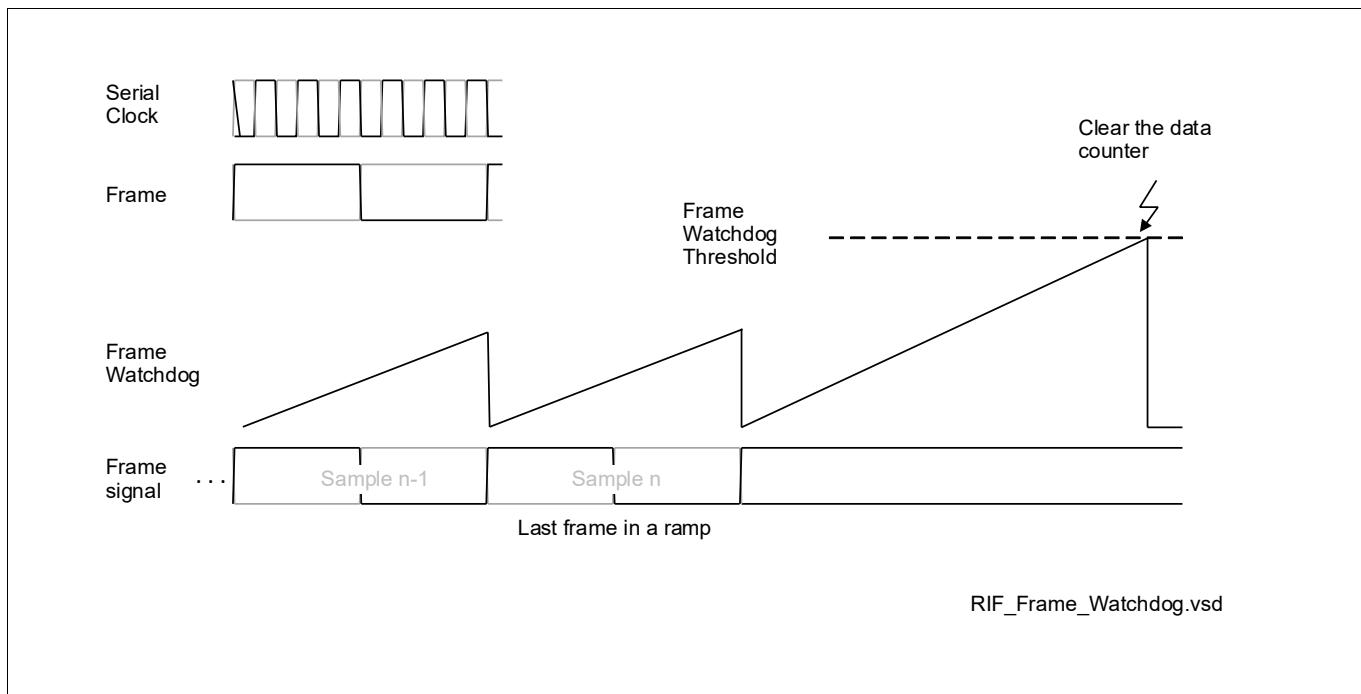
- CPU starts the RIF module
- CPU starts the external ADC by sending an SPI message
- CPU sends SPI command to start the ramp(s), the external ramp generator starts and when the ramp is in its valid part, the external ADC starts sending samples
- The external ADC completes the ramp by itself. In parallel the RIF module counts the samples and the ramps and raises an error in case of mismatch.

### 24.3.11.1 Frame Watchdog

Frame Watchdog is one 10-bit free running timer monitoring the frame signal and being reset by the incoming frames. It is driven by the RIF kernel frequency  $f_{ADAS}$ . It is configured to count for longer than the maximal serial frame length that appears in an application (CRC frame length if CRC is used, otherwise data frame length). If there is a pause in the data stream, the watchdog timer reaches the threshold. This is interpreted as an end of a ramp and the sample counter is reset to zero. The Frame Watchdog is started when receiving a frame, and stopped when the FWDG event has occurred, until the next frame is received.

Frame watchdog also monitors the data flow from the internal ADCs.

After reset, the default value of the register **FWDG.THRESHOLD** is 0. With this setting, the frame watchdog does not generate Ramp End interrupts or Frame Watchdog interrupts.

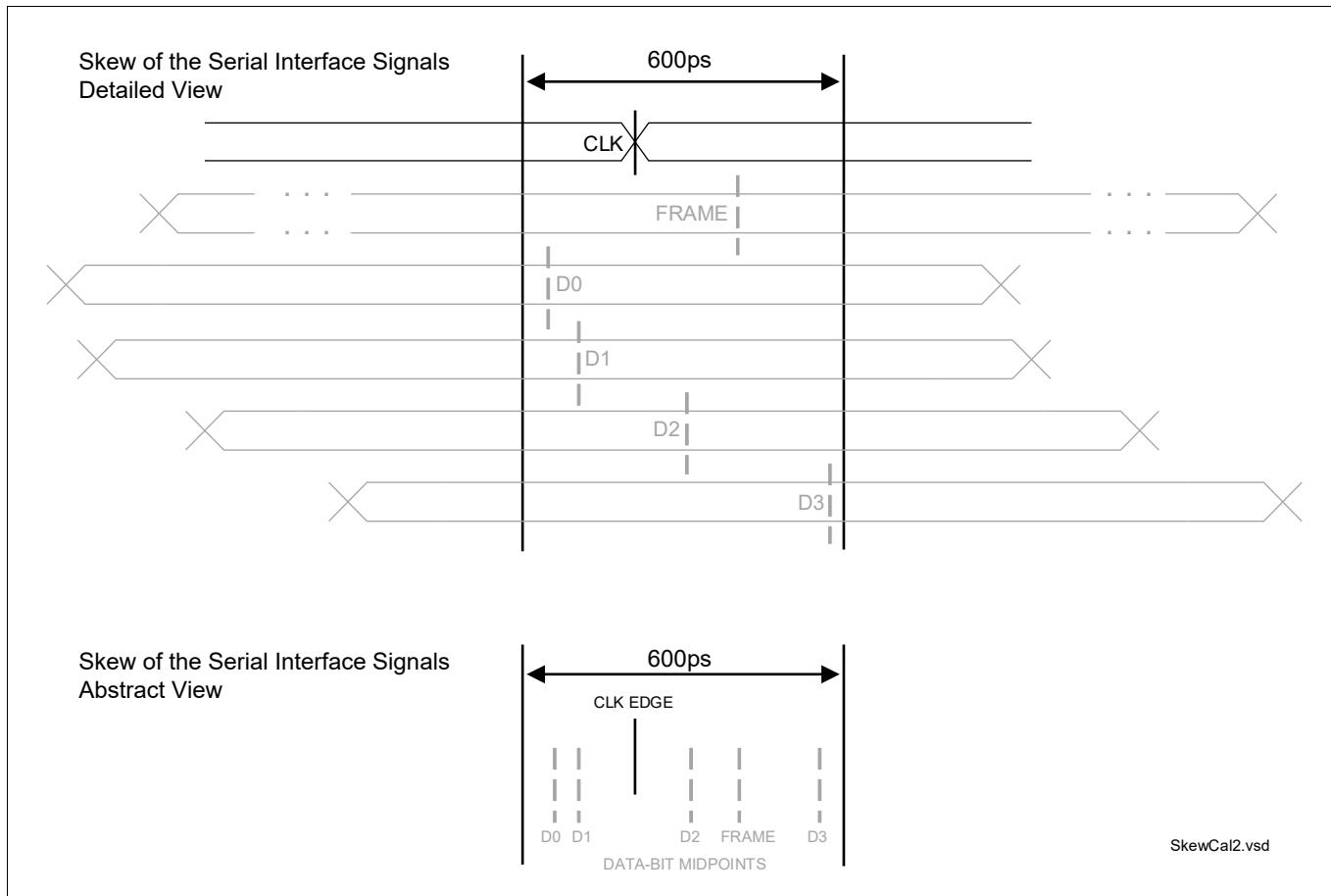
**Radar Interface (RIF)****Figure 310 Frame Watchdog Operation**

## Radar Interface (RIF)

### 24.3.11.2 On-Chip Signal Delay Calibration

To support TC3Ax data rates up to 600Mbit/s, a new skew calibration algorithm will be introduced.

Each SRIF quad comprises 4 LVDS data receivers supporting the IEEE 1596.3 standard with data rates up to 400MBit/s. This standard allows a skew of 600ps between any two LVDS input signals. A dedicated calibration mode allows to compensate for these skews.



**Figure 311 Overview of the Allowed Skew**

The skew calibration feature can calibrate the skew only for baud rates in the range of 200MBaud to 400MBaud. Do not use the skew calibration for baud rates below 200MBaud. Use the uncalibrated interface considering the timings defined in the data sheet.

The deserializer unit needs to calibrate the delay lines at least one time at the start of the application. The command for the external device to start the calibration is sent over the SPI interface.

During the initialization phase of the system the user just needs to trigger the skew compensation (Calibration Enable, bit **ESI.CALEN**) and wait till all serial channels have been compensated. The end of this sequence will be indicated to the user by special control signals (Calibration Busy, bit **ESI.CALBSY** and Calibration Status, bit **ESI.CALSTAT**).

The compensation is triggered by the user via SFRs but then steered entirely by a state machine within the RIF. To enable this feature it is necessary to set the external LVDS transmitter (radar front-end) into a special calibration mode via the SPI interface.

One calibration is necessary after system start before the interface is used. Calibration should be repeated if:

- Data seems corrupted, indicated by CRC-errors.
- A significant change in the operating environment of the system has been detected.

## Radar Interface (RIF)

### 24.3.11.3 On-chip Signal Delay Calibration Sequence

In order to perform delay calibration, the following sequence must be followed:

- Disable the deserializers by using the bits **IPI.EN0, 1, 2, 3**
- Initiate the calibration pattern transmission by the radar front-end, for example by sending a corresponding QSPI frame
- Start the calibration procedure by writing "1" to **ESI.CALEN**, and ensure that the calibration has been successful by monitoring the status bits **ESI.CALBSY** and **ESI.CALSTAT**.

When performing delay calibration after reset, the step 1 can be omitted, because the default value of the **IPI.ENx** bits is disabled.

When performing delay recalibration during run-time, it has to be done after a ramp has finished, by following the three steps above.

Afterwards, after successful calibration, the normal operation can be resumed by initiating the next ramp at the radar front-end.

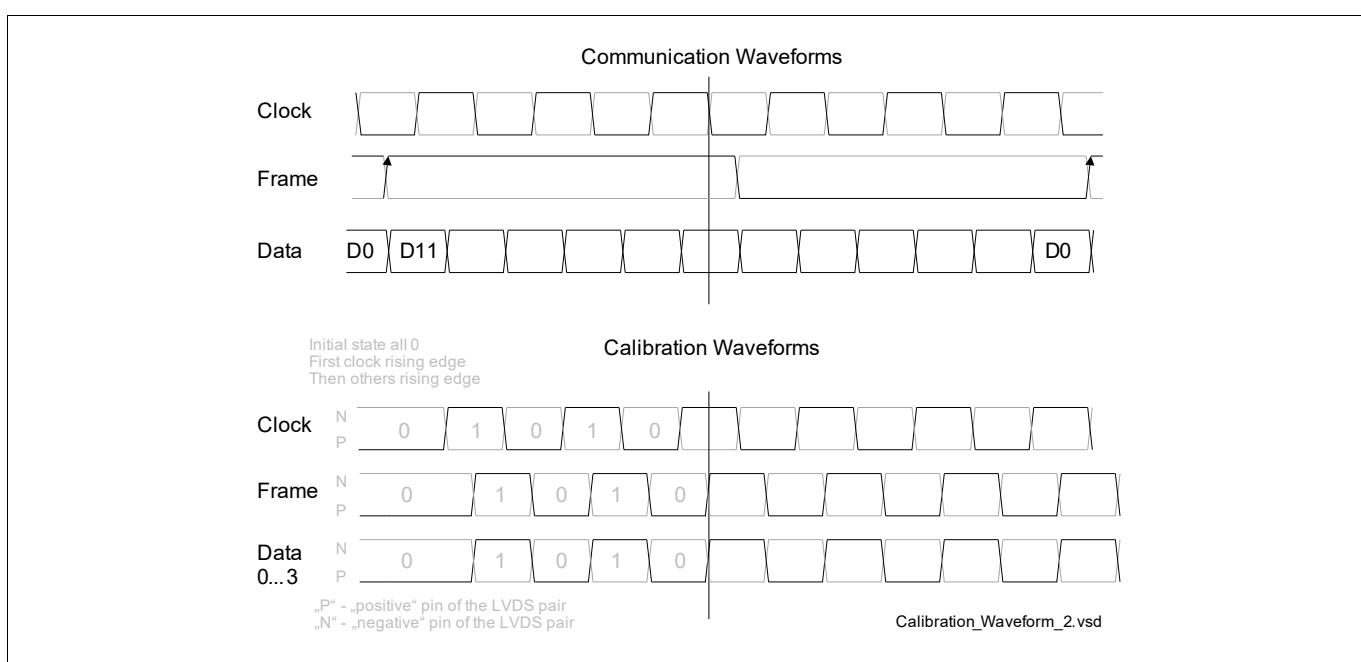
The total time allowed for calibration is 10us, SPI communication time not included.

The application software should check by polling the RIF module if the calibration sequence has finished within the time window as defined above, because heavily distorted signals on the serial lines can cause the calibration pattern to run without terminating.

**Note:** *If the application software activates an application reset for the RIF module while CCUCON5.ADASDIV=0<sub>B</sub>, the start-up software cannot update the default delay calibration value inside the RIF module. The application software must take care that CCUCON5.ADASDIV!=0 when triggering application reset and the RIF module is used afterwards.*

### 24.3.11.4 Waveforms Required to Perform On-Chip Signal Delay Calibration

**Figure 312** shows the optimal waveform for performing delay calibration of the RIF serial signals compared to the normal communications waveforms. All signals Clock, Frame and Data toggle with same frequency. Frame and Data rising edges are delayed relative to the Clock rising edge for a quarter of the toggle period.



**Figure 312 Delay Calibration Waveforms**

## Radar Interface (RIF)

The skew between Frame and Data signals must be within the specified skew limits of +/-600ps to ensure correct behaviour of the calibration circuit.

The register field RIF.ESI.CALBSY is set when calibration starts. The calibration is finished when the register field RIF.ESI.CALBSY is cleared. Successful completion of calibration is signaled by setting the register field RIF.ESI.CALSTAT.

The completion of calibration can also be signaled by interrupt. The interrupt is enabled by setting the RIF.INTCON.CALE register field. If an interrupt occurs, it can be determined if the cause was end of a calibration by reading the INTCON.CALF register field. This will be set if a "calibration end" event has occurred.

The calibration sequentially attempts to set to zero the skew on the signals

- FRAME (DATA0)
- DATA0
- FRAME (DATA1)
- DATA1
- FRAME (DATA2)
- DATA2
- FRAME (DATA3)
- DATA3

Calibration is only attempted for enabled data channels. An independent skew compensation is determined for each data channel for the single FRAME input to the MCU.

If, at any point in the calibration sequence, a channel remains uncalibrated even if the delay is set to maximum or minimum direction, the clock delay for the channel is adjusted by one step in the other direction (e.g. if the delay for the signal under calibration is at maximum, then the clock delay is adjusted one step towards minimum value) and the entire calibration process is restarted.

### 24.3.11.5 Delay Adjustment During Calibration

Compensation for skew is made using a block comprising seven, identical delay elements. The delay through each element varies with the operating point of the MCU .

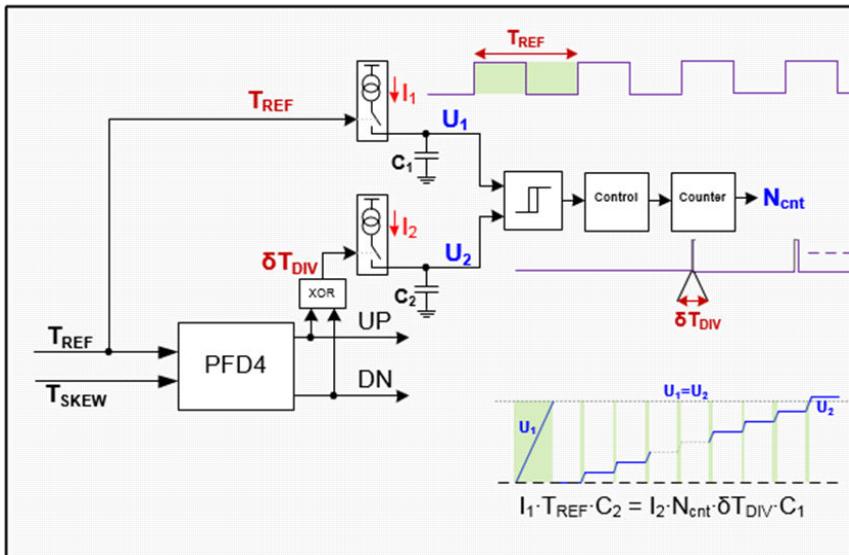
**Table 1138 Variation of Delay with Operating Point**

Corner	LSB Delay
Nominal	150ps
Slow Process/VDD <sub>Min</sub> /TEMP <sub>Max</sub>	240ps
Fast Process/VDD <sub>Max</sub> /TEMP <sub>Min</sub>	75ps

### 24.3.11.6 Skew Measurement During Calibration

Skew is measured by a charge measurement process. A reference level is established by using a current source to charge a capacitor for one period of the reference signal. A second signal is derived from the phase difference between the reference signal and the signal under test. The second signal is used to charge a second capacitor. The skew value is a count derived from the time needed for the second capacitor to reach the voltage stored on the first capacitor.

## Radar Interface (RIF)



**Figure 313 Skew Measurement schematic**

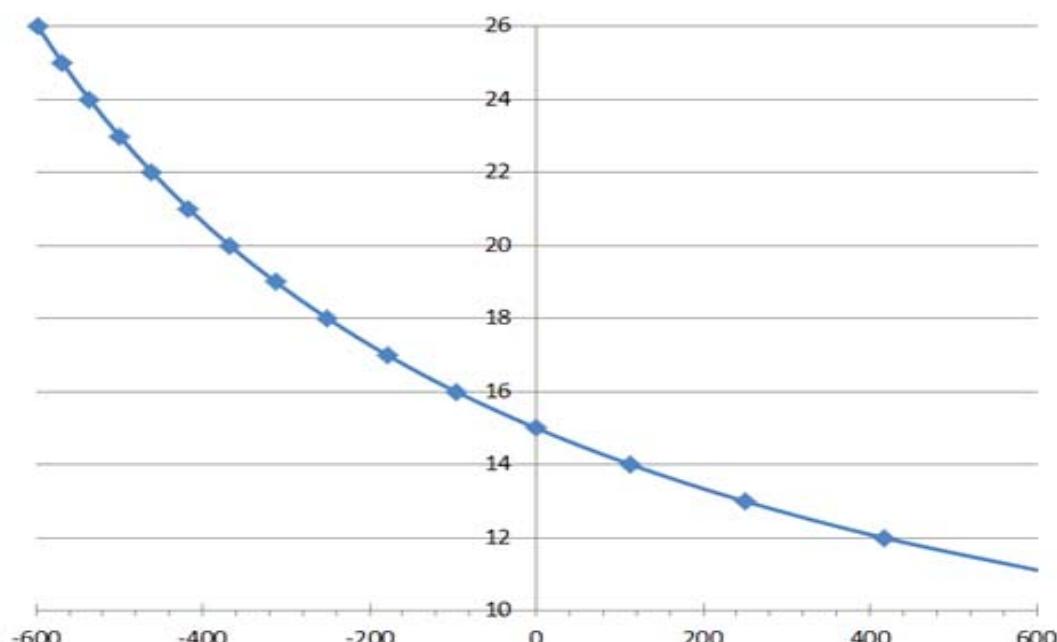
It's a consequence of this method that the relationship between the count and the skew is non-linear.

(24.1)

$$N_{CNT} = \frac{I_1 \times T_{REF} \times C_2}{\sigma T_{DIV} \times C_1 \times I_2}$$

The method is also prone to variation between devices caused by circuit mismatch. Calibration during production test to determine the reference count equivalent to zero skew is required.

The reference count is expected to vary around a value of 16 between limits of 14 and 18.



**Figure 314 Relationship between skew and count**

---

**Radar Interface (RIF)****24.3.11.7 Reference Skew Value and Error Limits**

The reference value and positive and negative error limits are determined at device calibration during production test and are stored in the UCB\_SSW table. They are transferred to the RIF.SKEWCAL register by system firmware and should not be modified by application software.

The UCB values are stored in the fields RIF\_SKEWCAL.ACCEP, RIF\_SKEWCAL.ACCEC and RIF\_SKEWCAL.VALUE

The RIF\_SKEWCAL.CALRESULT field contains the last measured skew value. This field is intended for use during calibration of the RIF skew measurement function at production test. It will only contain meaningful data after production test calibration.

The value read from this field during production test will be stored in the UCB\_SSW table.

*Note:* SKEWCAL register is not applicable for TC3Ax

**24.3.11.8 RAMP1 Signal**

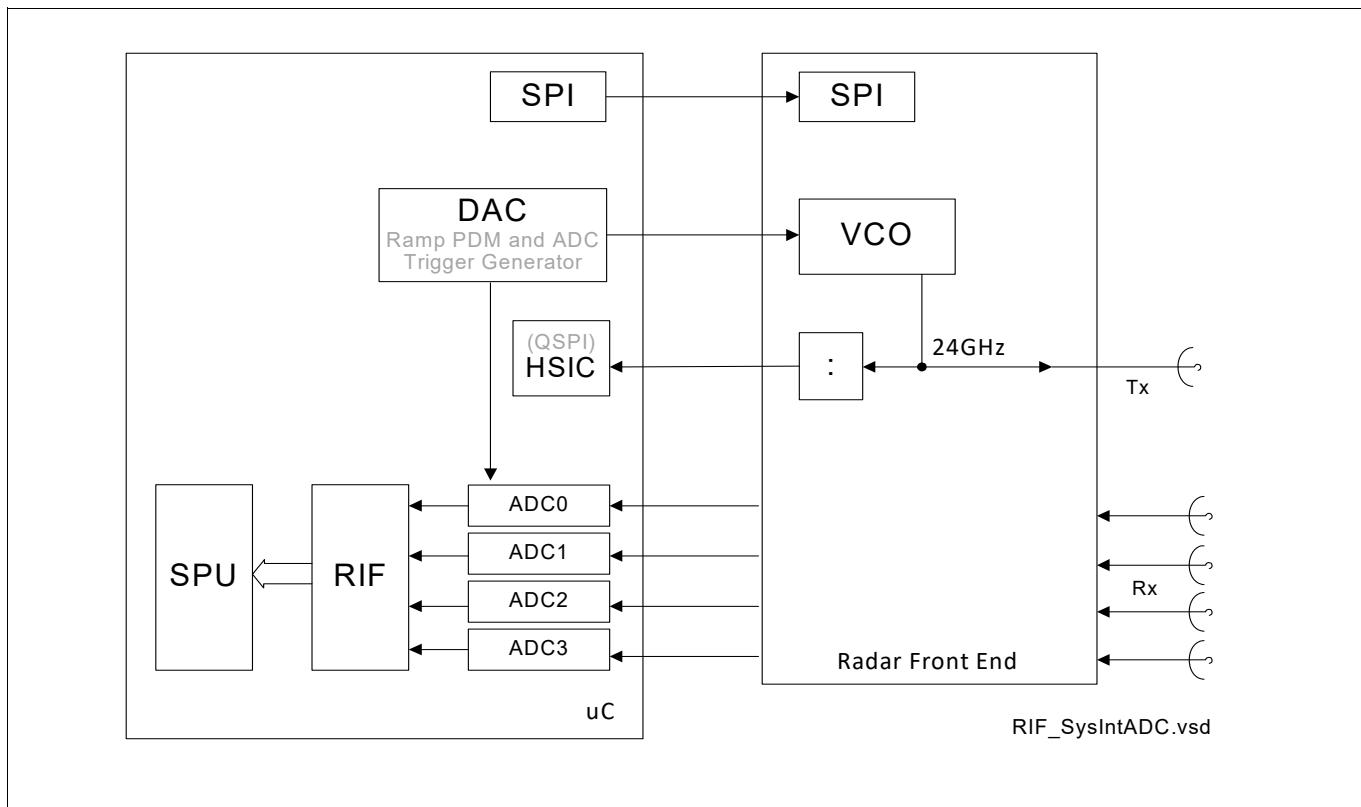
RAMP1 resets the sample counter at activation. A check is performed to verify that the programmed number of samples has been received at deactivation. If a mismatch is detected, the error interrupt is raised.

This feature is unavailable in TC3Ax.

## Radar Interface (RIF)

### 24.3.12 Internal ADCs Use-Case

In this use-case, the internal ADC is used for sampling the radar data.



**Figure 315 System View, Internal ADCs Use-Case**

The radar interface is an input only receiver for the data provided by the internal ADC. Control functions for the internal ADC have to be provided by other modules like HSPDM or timer modules.

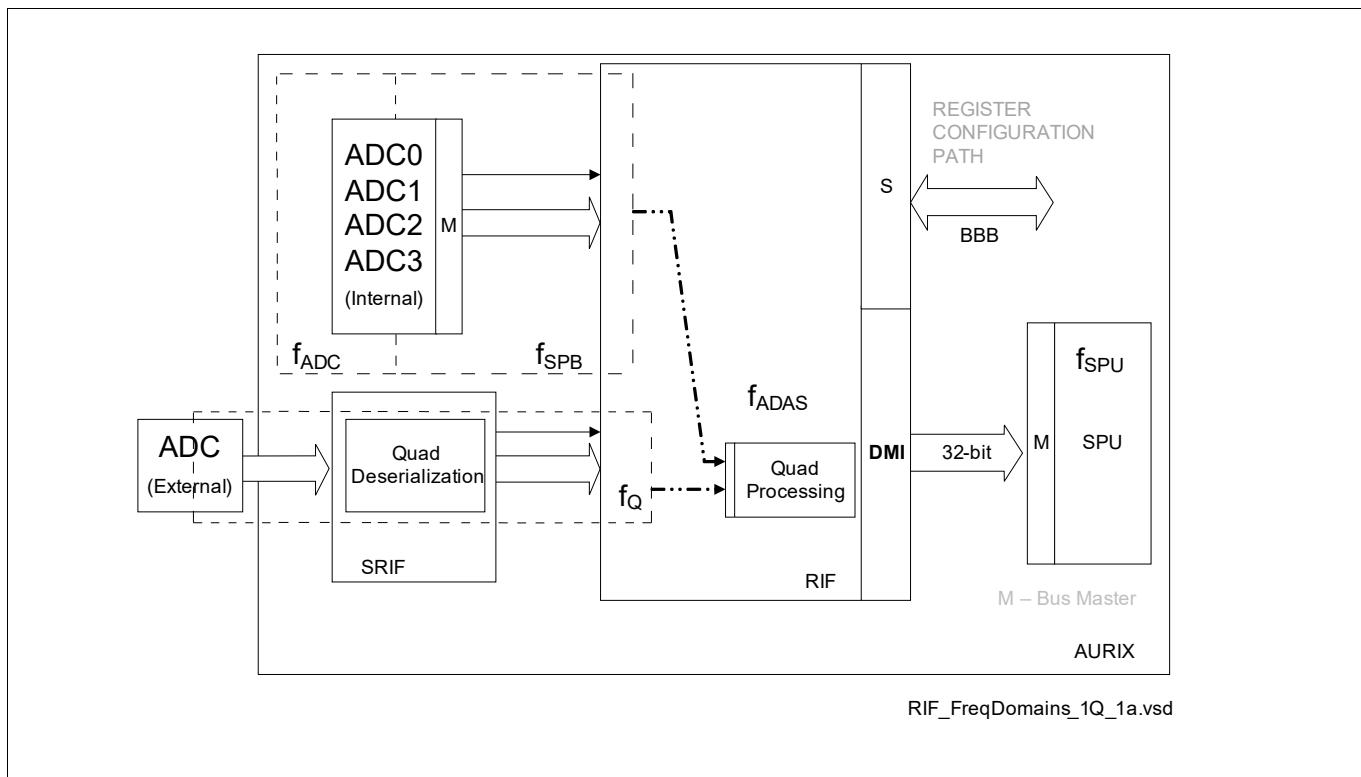
Internal ADC use-case uses the Frame Watchdog feature. It sends only valid samples with pauses between the ramps.

### 24.3.13 Frequency Domains

The RIF module contains several asynchronous frequency domains.

There are four internal ADCs that share the FIFO with Quad deserializer and support the same modes of lane management as the external ADCs:

## Radar Interface (RIF)



**Figure 316 Top View of the Frequency Domains**

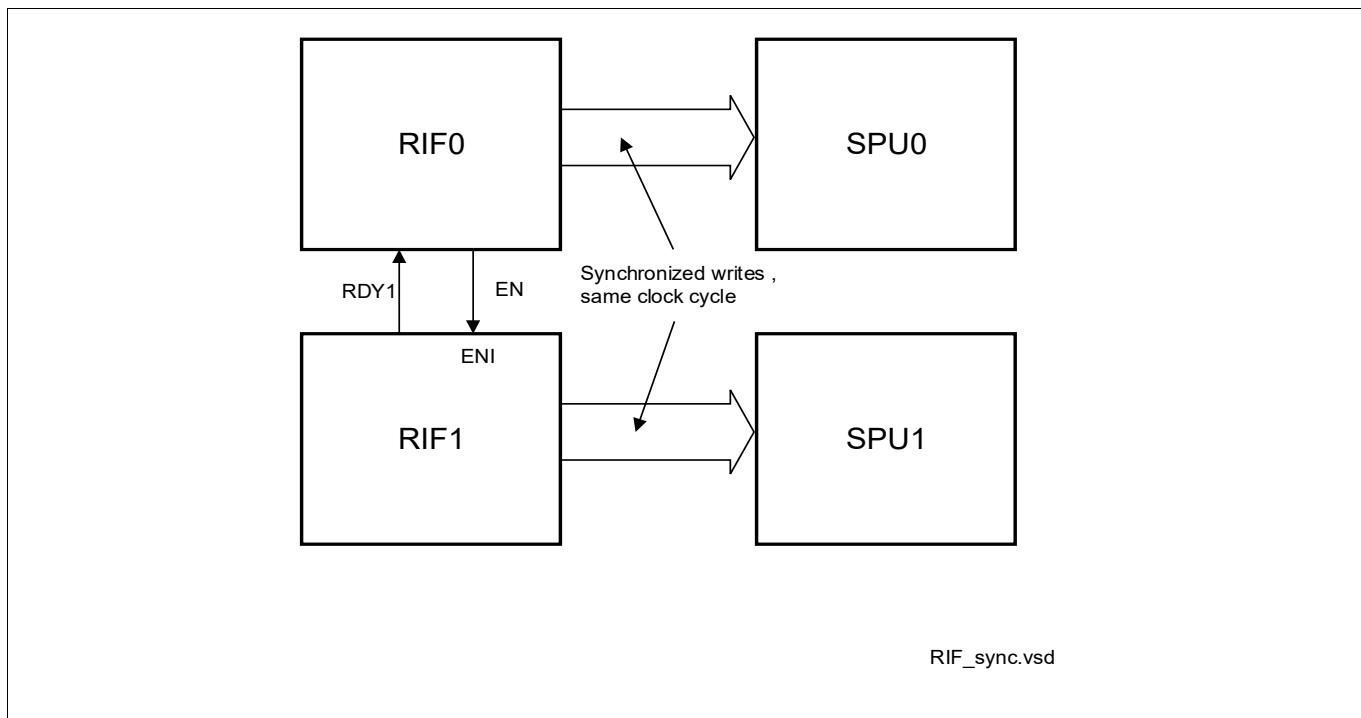
- Quad 0 frequency domain  $f_{Q0}$ , and internal ADC Quad  $f_{Q4}$
- Core frequency domain  $f_{ADAS}$
- ADC frequencies  $f_{ADC}$  and  $f_{SPB}$

### 24.3.13.1 Synchronization of two RIF Modules

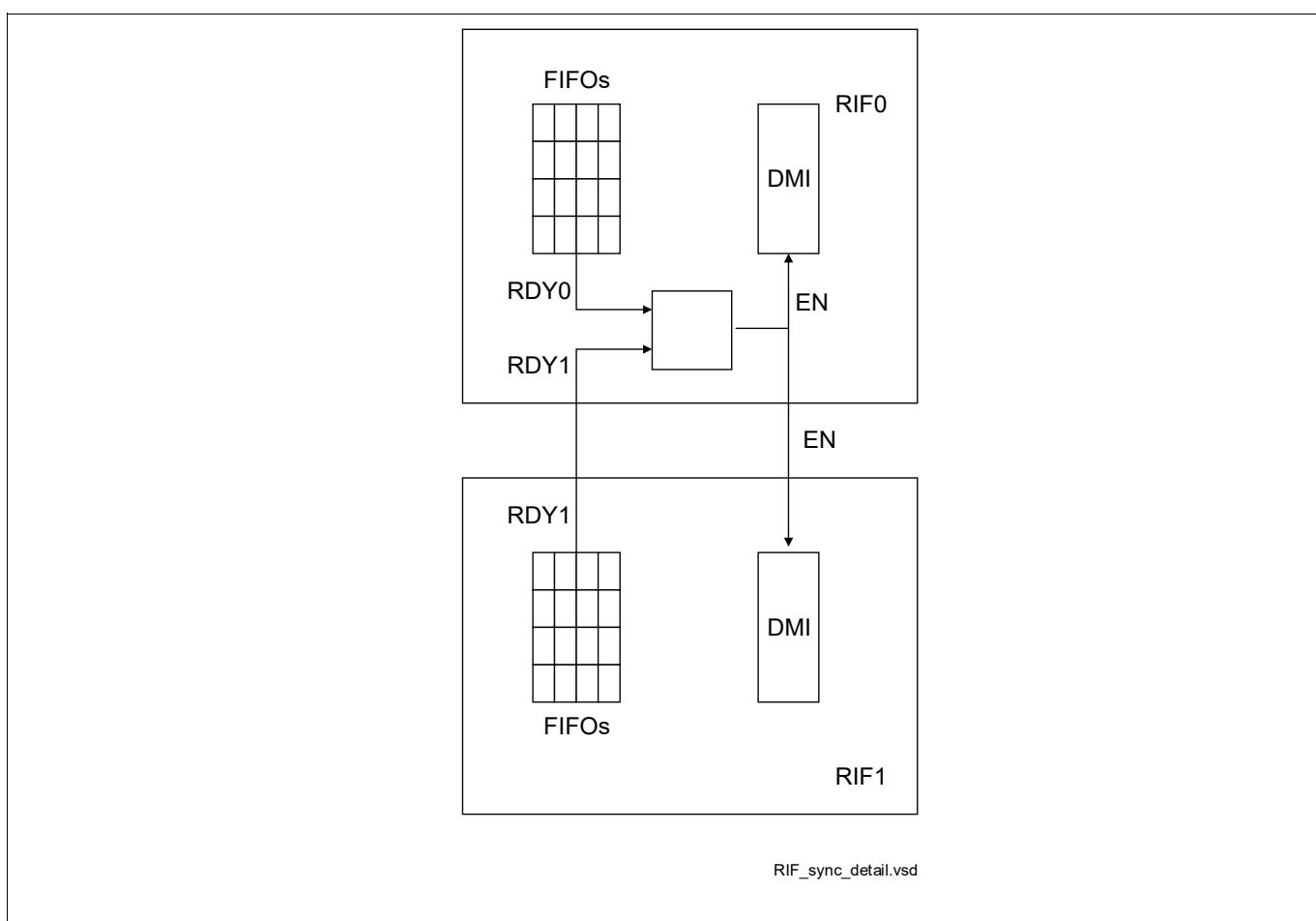
Two RIF modules can operate synchronously and provide their corresponding data to in the same clock cycle, enabling lockstep operation of the Signal Processing Units (SPUs). RIF synchronization is also necessary when two RIF modules communicate with one SPU. The RIF0 module takes the role of the “master”, generating from the ready signals RDY0 and RDY1 the enable signal EN for both RIF0 and RIF1 module.

Two RIF modules are capable of maintaining synchronization if the skew between the inputs of the two RIF instances is up to two frames long.

### Radar Interface (RIF)



**Figure 317 Synchronization of RIF Instances**



**Figure 318 Synchronization Architecture**

---

## Radar Interface (RIF)

### 24.3.13.2 Interrupts

There are two interrupt lines which are driven by several event sources. They are:

- ERR Interrupt
  - CRC Error Channel 0 - triggered on CRC Error on Channel 0
  - CRC Error Channel 1 - triggered on CRC Error on Channel 1
  - CRC Error Channel 2 - triggered on CRC Error on Channel 2
  - CRC Error Channel 3 - triggered on CRC Error on Channel 3
  - RAMP1 Error - triggered if at RAMP End event the number of received samples is not equal to the programmed value
- INT Interrupt
  - Calibration End - triggered after the Calibration Sequence of the input delay lines has been finished
  - Frame Watchdog Overflow - triggered when the watchdog timer value is about to exceed the threshold value
  - Ramp Start - triggered when the first sample of the new ramp has been received
  - Ramp End - triggered when the Frame Watchdog is about to exceed the threshold value or the RAMP1 signal goes inactive

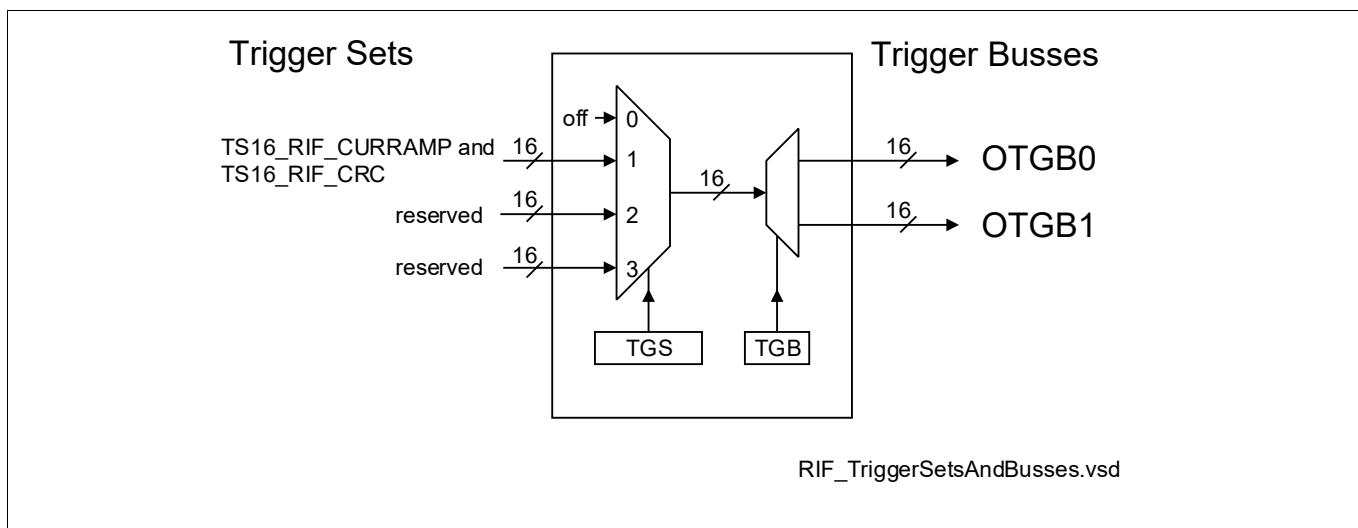
Each source have its own enable, flag, set and clear bit, see register **INTCON**.

## Radar Interface (RIF)

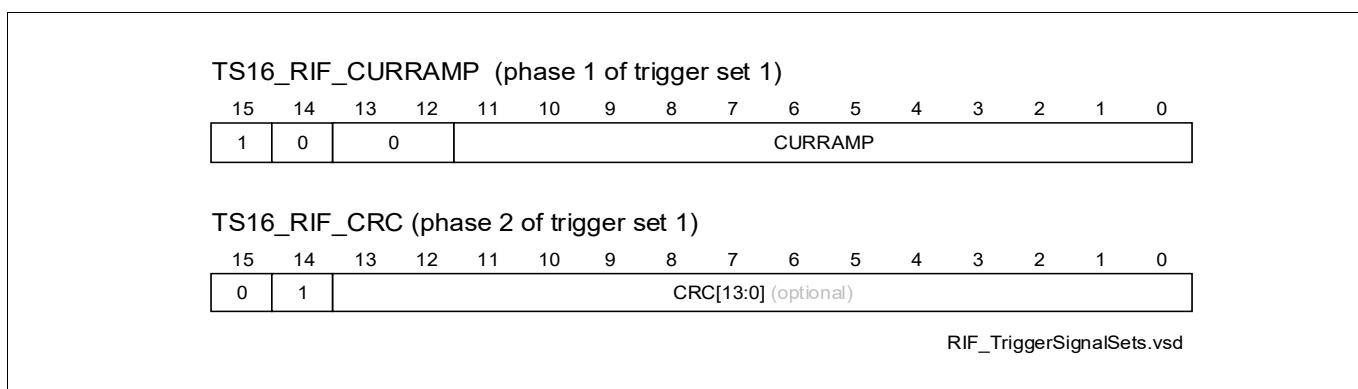
### 24.3.14 OCDS Trigger Sets

In order to support the debugging activities, the RIF module provides a set of internal signals to the on-chip debug system OCDS. An overview of this feature is shown in [Figure 319](#). Its configuration is done by using the bits **OCS.TGS** and **TGB**.

An edge on any module internal signal belonging to the selected set going out on one of the two OTGB busses triggers an action of the OCDS system.



**Figure 319 Overview of the Trigger Sets and Busses**



**Figure 320 Overview of the Trigger Signal Sets**

**Table 1139 RIF Trigger Sets**

Name	Description
TS16_RIF_CURRAMP	Counter value of the ramp counter
TS16_RIF_CRC	CRC0 value for the latest ramp

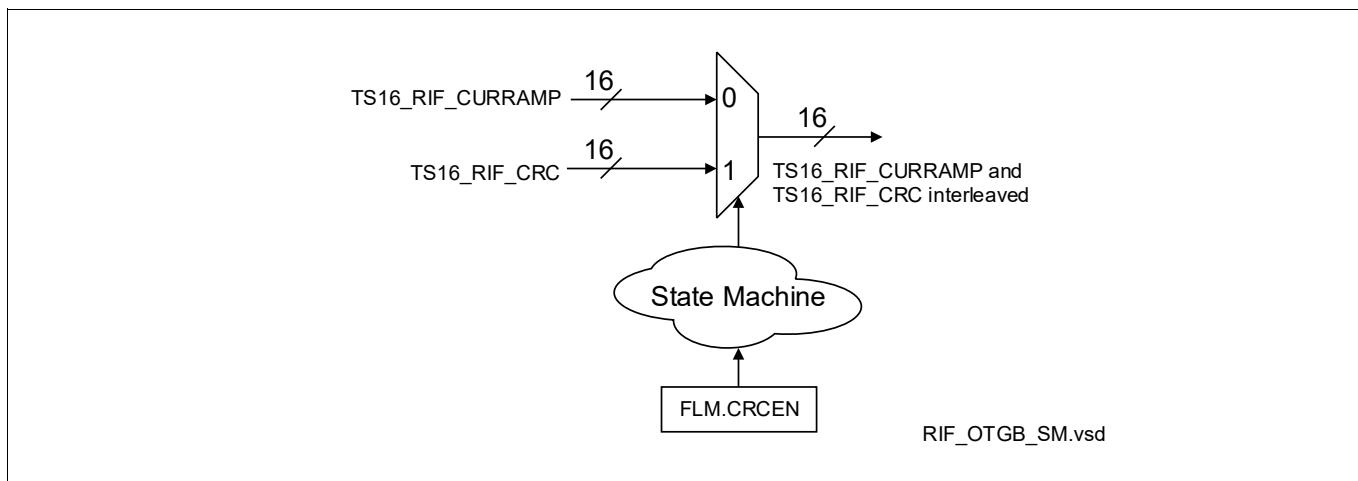
**Attention:** The value **CRC0** corresponds to **Channel 0** if no channel swapping is performed, see [FLM.FSWP](#) and [FLM.MODE](#). If swapping is enabled, the **CRC0** corresponds to the appropriate channel, see [Figure 300](#).

The trigger set 1 is used, the sets 2 and 3 are not used (padded with 0).

Set 1, two times 16 bit:

## Radar Interface (RIF)

- 12 bits: **RSM1.CURRAMP**
- 14 bits: CRC[13:0]. If the CRC is disabled, the first value **RSM1.CURRAMP** is output instead.



**Figure 321 Trigger Set State Machine**

The state machine toggles between TS16\_RIF\_CURRAMP and TS16\_RIF\_CRC if the CRC is enabled, or statically selects TS16\_RIF\_CURRAMP if the CRC is disabled. The toggling is done with the  $f_{SPB}$  frequency. The state machine is started when OCDS is enabled and a valid RIF Trigger Set is selected () .

### 24.3.15 Register CRC

The register CRC uses a 32-bit polynomial compatible with the TriCore CRC instruction and assumes that the register contents are converted to an LSB first, serial data stream.

- The RIF module generates the Register CRC using all the “rw” configuration bit-fields, from the registers **ESI** to **SFCON**
- The **BPI\_FPI Registers** and the **REGCRC** register itself are not part of the Register CRC
- The unused register address space between **DBGDLY1** and **DBG0** (address offsets  $005C_H$  to  $0007C_H$ ) is excluded from the CRC calculation (the addresses are skipped rather than the data being replaced by  $0000\_0000_H$ )
- The volatile “rh” bit-fields showing status data that are modified during run time by the RIF module itself are replaced by zeros
- The write-only “w” bit-fields return “0” on read and therefore are included in the Register CRC as zeros

When calculating the register CRC, the RIF module replaces the following bits by zeros:

- **ESI.CALEN**, **CALSTAT**, **CALBSY**
- **IPI.SDDV**
- **RSM1.CURRAMP**
- **RSM2.CURSAMPLE**
- **INTCON.[X]F** bits (i.e. all the flag bits... the ENABLES are left alone & passed to the CRC)
- Entire **FLAGSSET** register
- Entire **FLAGSCL** register
- Entire **RSM2CAP** register
- **SKEWCAL.R16**
- Entire **DBGDLY0** register

## Radar Interface (RIF)

- Entire **DBGDLY1** register
- Entire **DBG0** register
- Entire **DBG1** register

The CRC polynomial runs periodically once enabled and the CRC value generated from the register contents is compared with the value stored in the **REGCRC.CRC** bit-field. If the comparison fails, than an SMU alarm is generated.

The CRC-32 polynomial is compatible with the TriCore CRC-32 algorithm.

The Register CRC is enabled and disabled by writing to the Safety Function Register **FLM.REGCRCEN**.

The control and comparison logic of the safety mechanism is replicated using negative logic. Any difference in behavior between the two sets of logic results in an alarm.

The alarm can be tested writing an invalid CRC to **REGCRC.CRC**.

The register CRC mechanism operates in a round robin fashion, it takes a value of a register and writes it to the CRC engine each  $64 f_{ADAS}$  clock cycles.

Fault in safety mechanism is latent fault and is detected by BIST, which is performed each power on reset.

It is responsibility of the application software to stop the register CRC by using **FLM.REGCRCEN**, update the configuration of the RIF, write new CRC value in the CRC register, and enable the register CRC engine again, which restarts it from the initial state.

The application software has to write this bit field at the end of the configuration update. The CRC calculated by the software has to assume that this bit field has a value of “10b = enabled”.

### 24.3.16 Operating Modes

#### 24.3.16.1 Sleep Mode

The RIF module does not support sleep mode. See the **CLC.EDIS** bit-field description.

#### 24.3.16.2 OCDS Suspend Mode

The RIF module supports hard suspend. After hard suspend, the module should be reset. Soft suspend is not supported. See the **OCS.SUS** bit-field description.

### 24.3.17 Module Implementation

This section describes the product specific configuration of the RIF module instances with the clock control, port connections, interrupt control, and address decoding.

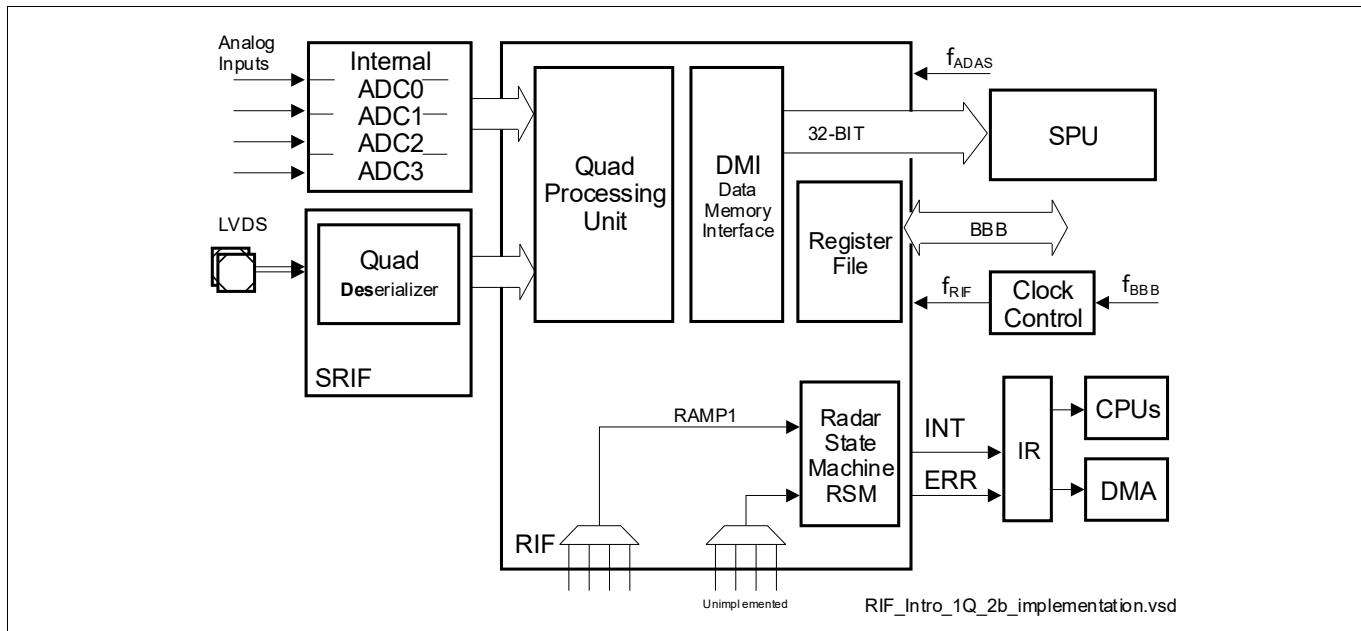
#### 24.3.17.1 ID Registers

The reset values of the RIF\_ID module identification registers are 00YY C0XX<sub>H</sub>.

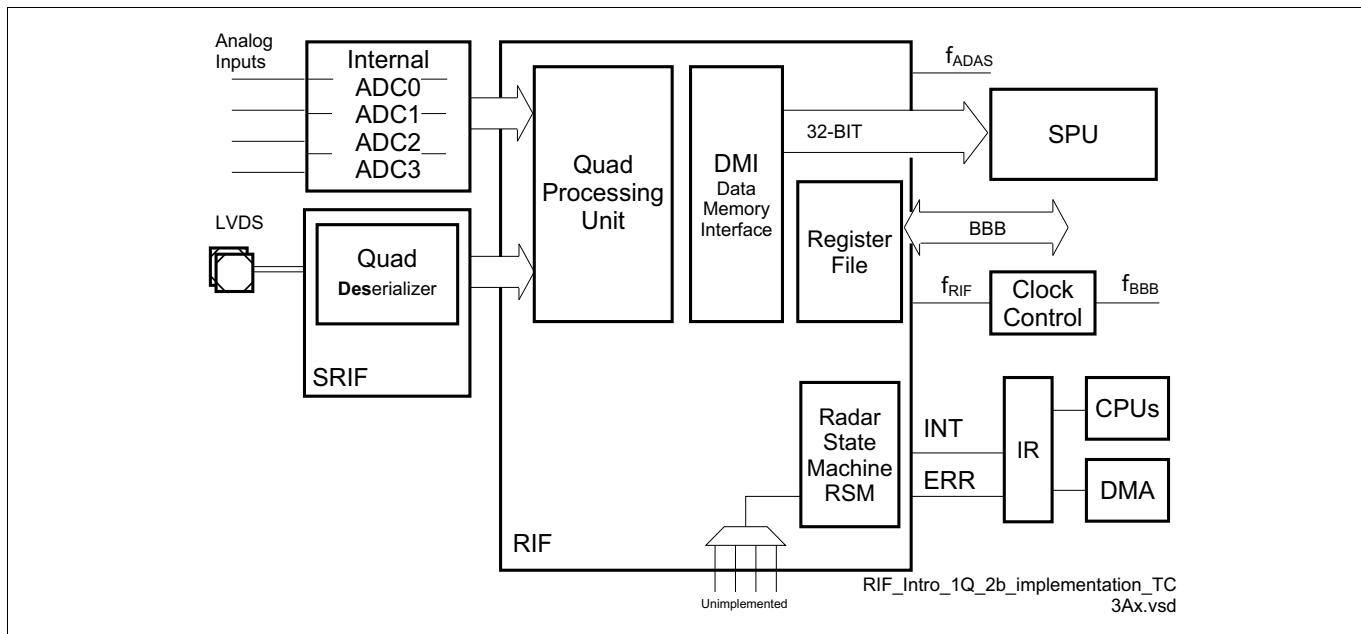
#### 24.3.17.2 Implementation Details

The following diagram shows the implementation details and interconnections of the RIF module.

### Radar Interface (RIF)



**Figure 322 RIF Implementation**



**Figure 323 RIF Implementation for TC3Ax**

#### 24.3.17.3 On-Chip Connections

The RIF module is connected to the LVDS pins, internal ADCs, and external pin(s) for the RAMP1 signal. See the pinning chapter for details.

##### 24.3.17.3.1 Connections to the internal ADCs

Each RIF module is connected to 4 internal ADCs, depending on the specific microcontroller derivative. The connections are listed in the corresponding derivative specification.

---

**Radar Interface (RIF)****24.3.17.3.2 RAMP1 Connections**

The RAMP1 signal is per default (reset value) low active. The unused inputs are connected to 1, the default inactive level. The default selected input is input 0, and it is always connected to default inactive 1.

The reset state of the RIF module is: Frame Watchdog enabled, RAMP1 connected to inactive level.

## Radar Interface (RIF)

### 24.4 Registers

This section describes the kernel registers of the RIF module. All RIF kernel register names described in this section will be referenced in other parts by the module name prefix “RIF\_”.

All kernel registers are initialised on application reset. All BPI registers belong to EEC reset except OCS, which belongs to debug reset (for definition see SCU section “Reset Operation”).

#### RIF Kernel Register Overview

Identification Register	Control Registers	Status Registers
ID	ESI IPI FLM DMI RSM0 RSM1 RSM2 INTCON FLAGSSET FLAGSCL FWDG DFU	SRIFOVRCFG RSM2CAP LVDSCON0 LVDSCON1 DBGDLY0 DBGDLY1 DBG0 DBG1 SFCON REGCRC

mca05790\_regs\_ov\_RIF.vsd

Figure 324 RIF Kernel Registers

**Radar Interface (RIF)****Table 1140 Register Overview - RIF (ascending Offset Address)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
CLC	Clock Control Register	0000 <sub>H</sub>	U,SV	SV,E,P	EEC Reset	<a href="#">69</a>
ID	Module Identification Register	0008 <sub>H</sub>	U,SV	BE	Application Reset	<a href="#">39</a>
ESI	External Serial Interface Register	0010 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">39</a>	<a href="#">39</a>
IPI	Internal Parallel Interface Register	0014 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">41</a>	<a href="#">41</a>
FLM	FIFO and Lane Management Register	0018 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">42</a>	<a href="#">42</a>
DMI	Data Memory Interface Register	001C <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">44</a>	<a href="#">44</a>
RSM0	Radar State Machine Register 0	0020 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">45</a>	<a href="#">45</a>
RSM1	Radar State Machine Register 1	0024 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">46</a>	<a href="#">46</a>
RSM2	Radar State Machine Register 2	0028 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">47</a>	<a href="#">47</a>
INTCON	Interrupt Control Register	002C <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">48</a>	<a href="#">48</a>
FLAGSSET	Flags Set Register	0030 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">51</a>	<a href="#">51</a>
FLAGSCL	Flags Clear Register	0034 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">53</a>	<a href="#">53</a>
FWDG	Frame Watchdog Register	0038 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">55</a>	<a href="#">55</a>
DFU	Data Formatting Unit Register	003C <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">55</a>	<a href="#">55</a>
SRIFOVRCFG	SRIF Override Configuration Register	0040 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">56</a>	<a href="#">56</a>
RSM2CAP	Radar State Machine 2 Capture Register	0044 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">57</a>	<a href="#">57</a>
SKEWCAL	Skew Calibration Register	0048 <sub>H</sub>	U,SV	U,SV,P,E	See page <a href="#">58</a>	<a href="#">58</a>
LVDSCON0	LVDS Control Register 0	004C <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">60</a>	<a href="#">60</a>
LVDSCON1	LVDS Control Register 1	0050 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">61</a>	<a href="#">61</a>
DBGDLY0	Debug Delay Register 0	0054 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">63</a>	<a href="#">63</a>
DBGDLY1	Debug Delay Register 1	0058 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">64</a>	<a href="#">64</a>
DLLCTL0	DLL Control Register 0	005C <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">65</a>	<a href="#">65</a>
DBG0	Debug Data Register 0	0080 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">66</a>	<a href="#">66</a>
DBG1	Debug Data Register 1	0084 <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">66</a>	<a href="#">66</a>
SFCON	Safety Functions Register	0088 <sub>H</sub>	U,SV	SV,P	See page <a href="#">67</a>	<a href="#">67</a>
REGCRC	Register CRC Register	008C <sub>H</sub>	U,SV	U,SV,P	See page <a href="#">68</a>	<a href="#">68</a>
OCS	OCDS Control and Status	00E8 <sub>H</sub>	U,SV	SV,P,OEN	See page <a href="#">70</a>	<a href="#">70</a>

**Radar Interface (RIF)****Table 1140 Register Overview - RIF (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
KRSTCLR	Kernel Reset Status Clear Register	00EC <sub>H</sub>	U,SV	SV,P	EEC Reset	<a href="#">74</a>
KRST1	Kernel Reset Register 1	00F0 <sub>H</sub>	U,SV	SV,P	EEC Reset	<a href="#">73</a>
KRST0	Kernel Reset Register 0	00F4 <sub>H</sub>	U,SV	SV,P	EEC Reset	<a href="#">72</a>
ACCEN1	Access Enable Register 1	00F8 <sub>H</sub>	U,SV	SV,SE	EEC Reset	<a href="#">72</a>
ACCENO	Access Enable Register 0	00FC <sub>H</sub>	U,SV	SV,SE	EEC Reset	<a href="#">71</a>

**List of Access Protection Abbreviations**

- U - User Mode
- SV - Supervisor Mode
- BE - Bus Error
- nBE - no Bus Error
- P - Access Protection, as defined by the ACCEN Register
- E - ENDINIT
- SE - Safety ENDINIT

## Radar Interface (RIF)

### 24.4.1 Kernel Registers

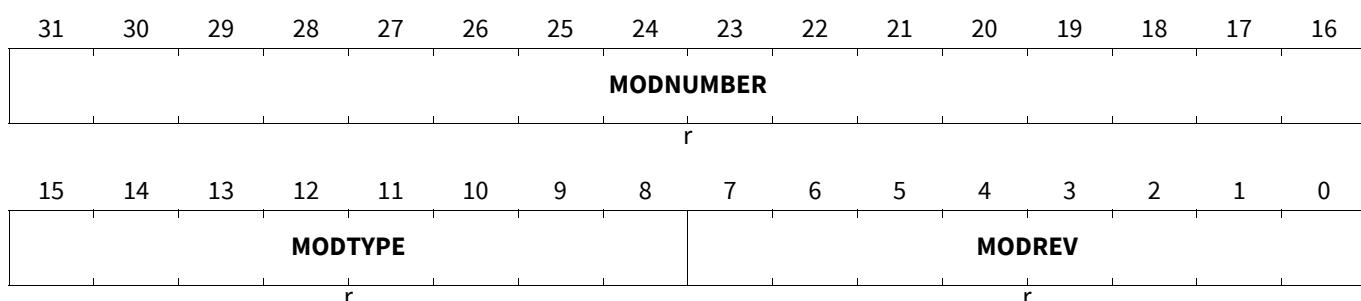
#### Module Identification Register

The Module Identification Register ID contains read-only information about the module version.

Current module identification number for Tc3Ax is 0x00e3c002 and for other derivatives it is 0x00e3c001.

##### ID

#### Module Identification Register (0008<sub>H</sub>) Application Reset Value: 00E3 COXX<sub>H</sub>



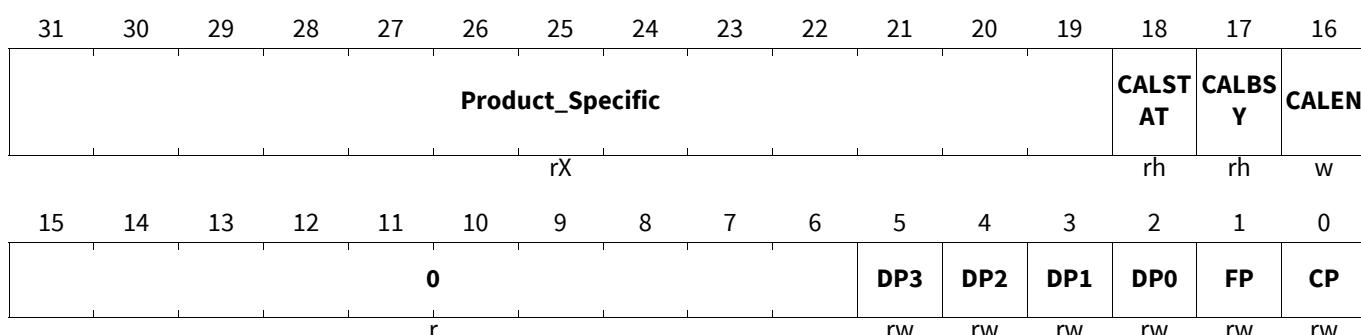
Field	Bits	Type	Description
<b>MODREV</b>	7:0	r	<b>Module Revision Number</b> MODREV defines the module revision number. The value of a module revision starts with 01 <sub>H</sub> (first revision).
<b>MODTYPE</b>	15:8	r	<b>Module Type</b> This bit field is C0 <sub>H</sub> . It defines a 32-bit module.
<b>MODNUMBER</b>	31:16	r	<b>Module Number Value</b> This bit field together with MODTYPE uniquely identifies a module.

#### External Serial Interface Register

The ESI register contains configuration parameters for the External Serial Interface.

##### ESI

#### External Serial Interface Register (0010<sub>H</sub>) Reset Value: Table 1141



## Radar Interface (RIF)

Field	Bits	Type	Description
CP	0	rw	<b>Clock Polarity</b> Defines the polarity of the clock signal on the clock input pins. 0 <sub>B</sub> default 1 <sub>B</sub> inverted
FP	1	rw	<b>Frame Polarity</b> Defines the polarity of the frame signal on the frame input pins 0 <sub>B</sub> default 1 <sub>B</sub> inverted
DP0	2	rw	<b>Data Polarity for Lane 0</b> Defines the polarity of the data signals on the data input pins 0 <sub>B</sub> default 1 <sub>B</sub> inverted
DP1	3	rw	<b>Data Polarity for Lane 1</b> Defines the polarity of the data signals on the data input pins 0 <sub>B</sub> default 1 <sub>B</sub> inverted
DP2	4	rw	<b>Data Polarity for Lane 2</b> Defines the polarity of the data signals on the data input pins 0 <sub>B</sub> default 1 <sub>B</sub> inverted
DP3	5	rw	<b>Data Polarity for Lane 3</b> Defines the polarity of the data signals on the data input pins 0 <sub>B</sub> default 1 <sub>B</sub> inverted
CALEN	16	w	<b>Calibration Enable</b> Enables the calibration mode of the deserializer and with its rising edge sets the CALBSY bit. In this state, the incoming bitstream is treated by the deserializer as calibration stream of 10101... and no parallel data is delivered to the digital part of the RIF. This bit is write only and returns 0 on read. 0 <sub>B</sub> write of zero - no effect 1 <sub>B</sub> write of one - brings the deserializer in calibration mode
CALBSY	17	rh	<b>Calibration Busy</b> Shows the current state of the deserializer, if a calibration or normal data reception is going on. 0 <sub>B</sub> no calibration ongoing 1 <sub>B</sub> calibration ongoing
CALSTAT	18	rh	<b>Calibration Status</b> Shows the status of the latest timing calibration sequence. The end of a calibration sequence is signalled by an interrupt, which is used by the CPU to check the status, and if OK, to switch the RIF to normal mode of operation. 0 <sub>B</sub> not OK (failed, signal paths uncalibrated) 1 <sub>B</sub> OK (successful)

## Radar Interface (RIF)

Field	Bits	Type	Description
<b>Product_Specifc</b>	31:19	rX	<b>Product_Specific</b> This bitfield is product specific.
<b>0</b>	15:6	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 1141 Reset Values of ESI**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

### Internal Parallel Interface Register

The IPI register contains configuration parameters for the Internal Parallel Interface.

#### IPI

#### Internal Parallel Interface Register (0014<sub>H</sub>) Reset Value: Table 1142

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>SDDV</b>	<b>DBGSEL</b>						<b>0</b>					<b>EN3</b>	<b>EN2</b>	<b>EN1</b>	<b>ENO</b>
rh	rw						r					rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							<b>0</b>					<b>PFP</b>	<b>0</b>	<b>DL</b>	
							r					rw	r	rw	

Field	Bits	Type	Description
<b>DL</b>	1:0	rw	<b>Data Length</b> Defines the data length of the ADC samples.  00 <sub>B</sub> 10 bits 01 <sub>B</sub> 12 bits 10 <sub>B</sub> 14 bits 11 <sub>B</sub> 16 bits
<b>PFP</b>	3	rw	<b>Parallel Frame Polarity</b> Defines the polarity of the frame (write signal from the internal ADCs) signal at the input of the RIF module. This is an internal MCU connection from the ADCs to the RIF module and is only used when data is being transferred from the internal ADCs. The interface is expected to function correctly with this field set to 0 <sub>B</sub> .  0 <sub>B</sub> default (latching on falling edge) 1 <sub>B</sub> inverted (latching on rising edge)

**Radar Interface (RIF)**

Field	Bits	Type	Description
<b>EN0</b>	16	rw	<b>Enable Deserializer 0</b> Enables / disables the deserializer 0. $0_B$ disabled $1_B$ enabled
<b>EN1</b>	17	rw	<b>Enable Deserializer 1</b> Enables / disables the deserializer 1. $0_B$ disabled $1_B$ enabled
<b>EN2</b>	18	rw	<b>Enable Deserializer 2</b> Enables / disables the deserializer 2. $0_B$ disabled $1_B$ enabled
<b>EN3</b>	19	rw	<b>Enable Deserializer 3</b> Enables / disables the deserializer 3. $0_B$ disabled $1_B$ enabled
<b>DBGSEL</b>	30:29	rw	<b>Selects the lane assigned to the registers DBG0 and DBG1</b> Selects the lane monitored by the registers DBG0 and DBG1. $00_B$ Lane 0 $01_B$ Lane 1 $10_B$ Lane 2 $11_B$ Lane 3
<b>SDDV</b>	31	rh	<b>Sample Debug Data Valid</b> Indicates if debug data is available in the DBG0 and DBG1 registers. $0_B$ No debug data available $1_B$ Debug data available in the DBG0 and DBG1 registers.
<b>0</b>	2, 15:4, 28:20	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 1142 Reset Values of IPI**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	Kernel Reset (software controlled by KRST0-1 registers)

**FIFO and Lane Management Register**

The FLM register contains configuration parameters for FIFO and Lane Management blocks.

## Radar Interface (RIF)

FLM

FIFO and Lane Management Register

(0018<sub>H</sub>)

Reset Value: Table 1143

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														REGCRCEN	
r															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CRCBS	EXPCR CWO	CRCER IN	CRCAL T	CRCE N	Product_Specific							FSWP	MODE	
r	rw	rw	rw	rw	rw	rX							rw	rw	

Field	Bits	Type	Description
MODE	0	rw	<b>FLM Mode</b> 0 <sub>B</sub> no swap of the data paths 1 <sub>B</sub> swap of data paths of channels 0-1 and 2-3
FSWP	1	rw	<b>Full Swap</b> 0 <sub>B</sub> no swap of the data paths 1 <sub>B</sub> swap of data paths of channels 0-3 and 1-2
Product_Specific	7:2	rX	<b>Product_Specific</b> This bitfield is product specific.
CRCEN	8	rw	<b>CRC Enable</b> Enables the CRC checking of the input data. 0 <sub>B</sub> Disabled 1 <sub>B</sub> Enabled
CRCALT	9	rw	<b>Alternative CRC</b> Select if the CRC calculation algorythm is the default or the alternative one. 0 <sub>B</sub> default CRC 1 <sub>B</sub> alternative CRC
CRCERIN	10	rw	<b>CRC Error Injection</b> Inject a deliberate error into the mechanism checking the CRC of RIF data to test the error generation. When set to ON, all RIF CRC checks should fail until this field is written with OFF. 0 <sub>B</sub> OFF, No Error Injected 1 <sub>B</sub> ON, Error Injected
EXPCRCWO	11	rw	<b>Expected CRC Word Order</b> Selects the expected order of the two 16-bit serial frames containing the CRC value generated by the radar front-end. 0 <sub>B</sub> Most Significant Word first 1 <sub>B</sub> Least Significant Word first
CRCBS	12	rw	<b>CRC Byte Swap</b> Selects if the CRC calculation algorythm swaps the bytes of an ADC sample before performing the CRC calculation. 0 <sub>B</sub> no swap (default) 1 <sub>B</sub> swap

## Radar Interface (RIF)

Field	Bits	Type	Description
<b>REGCRCEN</b>	17:16	rw	<b>Enable Bit for the Register CRC</b> $00_B$ invalid (and disabled) $01_B$ disabled (default) $10_B$ enabled $11_B$ invalid (and enabled)
<b>0</b>	15:13, 31:18	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 1143 Reset Values of FLM**

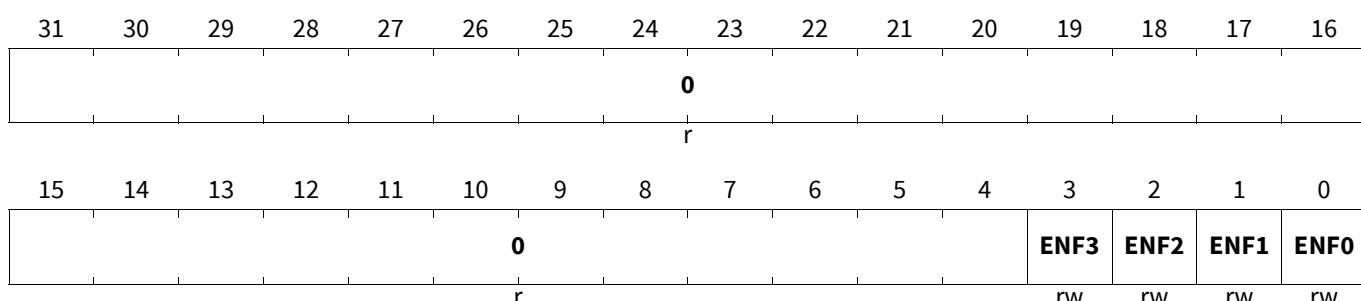
Reset Type	Reset Value	Note
Application Reset	$0001\ 0000_H$	
Kernel Reset (software controlled by KRST0-1 registers)	$0001\ 0000_H$	Kernel Reset (software controlled by KRST0-1 registers)

## Data Memory Interface Register

The DMI register contains the configuration parameters for the Data Memory Interface block.

### DMI

#### Data Memory Interface Register (001C<sub>H</sub>) Reset Value: Table 1144



Field	Bits	Type	Description
<b>ENFO</b>	0	rw	<b>Enable FIFO0</b> Provides possibility for dynamically starting and stopping the data stream $0_B$ disabled $1_B$ enabled
<b>ENF1</b>	1	rw	<b>Enable FIFO1</b> Provides possibility for dynamically starting and stopping the data stream $0_B$ disabled $1_B$ enabled

**Radar Interface (RIF)**

Field	Bits	Type	Description
<b>ENF2</b>	2	rw	<b>Enable FIFO2</b> Provides possibility for dynamically starting and stopping the data stream $0_B$ disabled $1_B$ enabled
<b>ENF3</b>	3	rw	<b>Enable FIFO3</b> Provides possibility for dynamically starting and stopping the data stream $0_B$ disabled $1_B$ enabled
<b>0</b>	31:4	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 1144 Reset Values of DMI**

Reset Type	Reset Value	Note
Application Reset	$0000\ 0000_H$	
Kernel Reset (software controlled by KRST0-1 registers)	$0000\ 0000_H$	Kernel Reset (software controlled by KRST0-1 registers)

**Radar State Machine Register 0**

The RSM0 register contains configuration parameters for the Radar State Machine.

**RSM0****Radar State Machine Register 0****(0020<sub>H</sub>)****Reset Value: Table 1145**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>INTAD</b>	<b>LCKST</b>							<b>0</b>							
<b>C</b>	<b>P</b>								<b>r</b>						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								<b>0</b>							

Field	Bits	Type	Description
<b>LCKSTP</b>	30	rw	<b>Lockstep Enable Bit</b> Enables synchronous delivery of ADC samples from two RIFs to two SPUs in case two RIF instances are available and used.
<b>INTADC</b>	31	rw	<b>Internal ADC Enable Bit</b> Defines if the radar interface accepts input from the internal or external ADCs. $0_B$ External ADC, deserializer enabled. $1_B$ Internal ADC, deserializer disabled independently of IPI.ENx.

## Radar Interface (RIF)

Field	Bits	Type	Description
0	29:0	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 1145 Reset Values of RSM0**

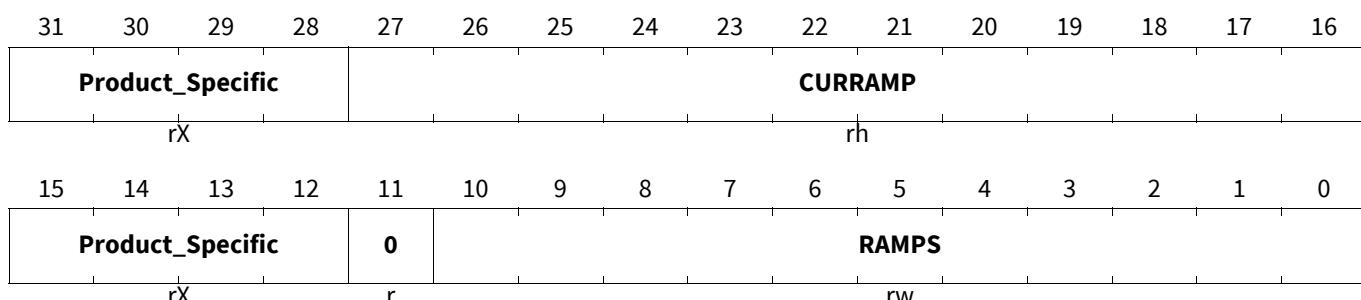
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	

## Radar State Machine Register 1

The RSM1 register contains configuration parameters for the Radar State Machine.

### RSM1

**Radar State Machine Register 1** **Reset Value: Table 1146**



Field	Bits	Type	Description
<b>RAMPS</b>	10:0	rw	<b>Number of Ramps per Chirp</b> Number of ramps in the range of 1 to 2048. ... 000 <sub>H</sub> 1 001 <sub>H</sub> 2 002 <sub>H</sub> 3 7FF <sub>H</sub> 2048
<b>Product_Specific</b>	15:12, 31:28	rX	<b>Product_Specific</b> This bitfield is product specific.

**Radar Interface (RIF)**

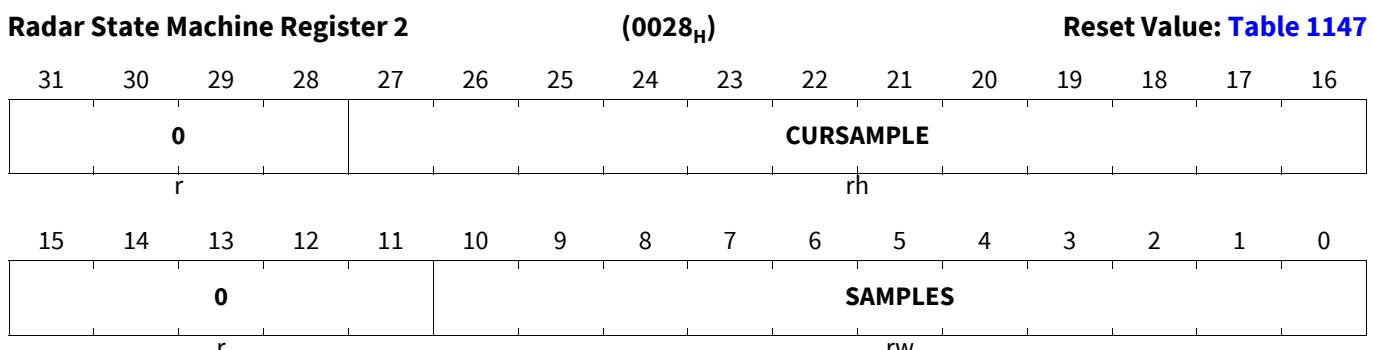
Field	Bits	Type	Description
<b>CURRAMP</b>	27:16	rh	<p><b>Number of Current Ramp</b></p> <p>The current ramp is incremented when each End of Ramp event occurs. When the counter value reaches RSM1.RAMPS or when a RAMP1 error occurs, the counter value is reset to 0.</p> <p>Number of ramps in the range of 0 to 2048.</p> <p>... ...</p> <p><math>000_H</math> 0  <math>001_H</math> 1  <math>002_H</math> 2  <math>800_H</math> 2048  <b>others</b>, reserved</p>
<b>0</b>	11	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 1146 Reset Values of RSM1**

Reset Type	Reset Value	Note
Application Reset	$0000\ 0000_H$	
Kernel Reset (software controlled by KRST0-1 registers)	$0000\ 0000_H$	

**Radar State Machine Register 2**

The RSM2 register contains configuration parameters for the Radar State Machine.

**RSM2**

Field	Bits	Type	Description
<b>SAMPLES</b>	10:0	rw	<p><b>Number of Valid Data Samples</b></p> <p>Range of 1 to 2048 samples.</p> <p>... ...</p> <p><math>000_H</math> 1  <math>001_H</math> 2  <math>002_H</math> 3  <math>7FF_H</math> 2048</p>

## Radar Interface (RIF)

Field	Bits	Type	Description
<b>CURSAMPLE</b>	27:16	rh	<p><b>Number of the Current Valid Data Sample</b></p> <p>This bitfield is incremented upon receiving each frame. When the counter value reaches RSM2.SAMPLES or when End of Ramp event occurs, the counter value is reset to 0.</p> <p>Range of 0 to 2048 samples.</p> <p>...</p> <p>000<sub>H</sub> 0 001<sub>H</sub> 1 002<sub>H</sub> 2 800<sub>H</sub> 2048 <b>others</b>, reserved</p>
<b>0</b>	15:11, 31:28	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Table 1147 Reset Values of RSM2**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	

**Interrupt Control Register**

The INTCON register contains enable and flag bits for the RIF interrupt sources.

Note that the flag bits are only set if the corresponding event triggers an interrupt. If the event occurs without the interrupt being enabled by the corresponding enable bit field, then the flag will not be set

**INTCON****Interrupt Control Register (002C<sub>H</sub>) Reset Value: Table 1148**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LOCKI F</b>	<b>REGC RCF</b>	<b>SMCF</b>		<b>0</b>	<b>R1SF</b>	<b>Produ ct_Sp ecific</b>	<b>R1EF</b>	<b>CRCF3</b>	<b>CRCF2</b>	<b>CRCF1</b>	<b>CRCF0</b>	<b>Produ ct_Sp ecific</b>	<b>REF</b>	<b>FWF</b>	<b>CALF</b>
rh	rh	rh		r	rh	rX	rh	rh	rh	rh	rh	rX	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					<b>R1SE</b>	<b>Produ ct_Sp ecific</b>	<b>R1EE</b>	<b>CRCE3</b>	<b>CRCE2</b>	<b>CRCE1</b>	<b>CRCE0</b>	<b>Produ ct_Sp ecific</b>	<b>REE</b>	<b>FWE</b>	<b>CALE</b>
					rw	rX	rw	rw	rw	rw	rw	rX	rw	rw	rw

## Radar Interface (RIF)

Field	Bits	Type	Description
<b>CALE</b>	0	rw	<b>Calibration End Interrupt Enable</b> Enables the interrupt on the set event of the corresponding flag. 0 <sub>B</sub> Disabled 1 <sub>B</sub> Enabled
<b>FWE</b>	1	rw	<b>Frame Watchdog Enable</b> Enables the interrupt on the set event of the corresponding flag. 0 <sub>B</sub> Disabled 1 <sub>B</sub> Enabled
<b>REE</b>	2	rw	<b>Ramp End Enable</b> Enables the interrupt on the set event of the corresponding flag. 0 <sub>B</sub> Disabled 1 <sub>B</sub> Enabled
<b>Product_Specific</b>	3, 9, 19, 25	rX	<b>Product_Specific</b> This bitfield is product specific.
<b>CRCE0</b>	4	rw	<b>CRC Error Flag Enable 0</b> Enables the interrupt on the set event of the corresponding flag. 0 <sub>B</sub> Disabled 1 <sub>B</sub> Enabled
<b>CRCE1</b>	5	rw	<b>CRC Error Flag Enable 1</b> Enables the interrupt on the set event of the corresponding flag. 0 <sub>B</sub> Disabled 1 <sub>B</sub> Enabled
<b>CRCE2</b>	6	rw	<b>CRC Error Flag Enable 2</b> Enables the interrupt on the set event of the corresponding flag. 0 <sub>B</sub> Disabled 1 <sub>B</sub> Enabled
<b>CRCE3</b>	7	rw	<b>CRC Error Flag Enable 3</b> Enables the interrupt on the set event of the corresponding flag. 0 <sub>B</sub> Disabled 1 <sub>B</sub> Enabled
<b>R1EE</b>	8	rw	<b>RAMP1 Error Enable</b> Enables the interrupt on the set event of the corresponding flag. 0 <sub>B</sub> Disabled 1 <sub>B</sub> Enabled
<b>R1SE</b>	10	rw	<b>RAMP1 Start Enable</b> Enables the interrupt on the set event of the corresponding flag. 0 <sub>B</sub> Disabled 1 <sub>B</sub> Enabled
<b>CALF</b>	16	rh	<b>Calibration End Interrupt Flag</b> Set after the calibration procedure has been finished. 0 <sub>B</sub> Event did not occur 1 <sub>B</sub> Event occurred

**Radar Interface (RIF)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>FWF</b>	17	rh	<b>Frame Watchdog Interrupt Flag</b> Set at frame watchdog overflow. 0 <sub>B</sub> Event did not occur 1 <sub>B</sub> Event occurred
<b>REF</b>	18	rh	<b>Ramp End Flag</b> Set either when the RAMP1 signal goes inactive or when the Frame Watchdog is about to exceed the programmed threshold value. If the CRC feature is enabled, the flag is set after the CRC frames have been received. 0 <sub>B</sub> Event did not occur 1 <sub>B</sub> Event occurred
<b>CRCF0</b>	20	rh	<b>CRC Error Flag 0</b> Set at CRC error on lane 0. 0 <sub>B</sub> Event did not occur 1 <sub>B</sub> Event occurred
<b>CRCF1</b>	21	rh	<b>CRC Error Flag 1</b> Set at CRC error on lane 1. 0 <sub>B</sub> Event did not occur 1 <sub>B</sub> Event occurred
<b>CRCF2</b>	22	rh	<b>CRC Error Flag 2</b> Set at CRC error on lane 2. 0 <sub>B</sub> Event did not occur 1 <sub>B</sub> Event occurred
<b>CRCF3</b>	23	rh	<b>CRC Error Flag 3</b> Set at CRC error on lane 3. 0 <sub>B</sub> Event did not occur 1 <sub>B</sub> Event occurred
<b>R1EF</b>	24	rh	<b>RAMP1 Error Flag</b> Set at RAMP1 Error Event. 0 <sub>B</sub> Event did not occur 1 <sub>B</sub> Event occurred
<b>R1SF</b>	26	rh	<b>RAMP1 Start Flag</b> Set at RAMP1 Start Event. 0 <sub>B</sub> Event did not occur 1 <sub>B</sub> Event occurred
<b>SMCF</b>	29	rh	<b>Safety Mechanism Control Flag</b> Signals invalid "00" or "11" setting of the-bit fields SPUCRCEN, LOCKIEN and REGCRCEN. 0 <sub>B</sub> 0 Event did not occur 1 <sub>B</sub> 1 Event occurred
<b>REGCRCF</b>	30	rh	<b>REGCRC Alarm Flag</b> Set at REGCRC error and alarm triggered. 0 <sub>B</sub> 0 Event did not occur 1 <sub>B</sub> 1 Event occurred

## Radar Interface (RIF)

Field	Bits	Type	Description
<b>LOCKIF</b>	31	rh	<b>RIF Internal Lockstep Alarm Flag</b> Set at internal lockstep error and alarm triggered. $0_B$ 0 Event did not occur $1_B$ 1 Event occurred
<b>0</b>	15:11, 28:27	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 1148 Reset Values of INTCON**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	

### Flags Set Register

The Flags Set Register contains set bits for the flag bits from the INTCON register.

#### FLAGSSET

(0030 <sub>H</sub> )																Reset Value: Table 1149			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16				
0								R1SS	Produ ct_Sp ecific	R1ES	CRCS3	CRCS2	CRCS1	CRCS0	Produ ct_Sp ecific	RES	FWS	CALS	
r	w	w	rX	w	w	w	w	w	w	w	w	w	w	w	rX	w	w	w	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
0																r			

Field	Bits	Type	Description
<b>CALS</b>	16	w	<b>Calibration End Flag Set</b> This is write only bit. Writing 0 has no effect. Writing 1 triggers an interrupt and sets the corresponding flag bit. $0_B$ No action $1_B$ Set the flag and trigger an interrupt
<b>FWS</b>	17	w	<b>Frame Watchdog Flag Set</b> This is write only bit. Writing 0 has no effect. Writing 1 triggers an interrupt and sets the corresponding flag bit. $0_B$ No action $1_B$ Set the flag and trigger an interrupt

## Radar Interface (RIF)

Field	Bits	Type	Description
<b>RES</b>	18	w	<b>Ramp End Set</b> This is write only bit. Writing 0 has no effect. Writing 1 triggers an interrupt and sets the corresponding flag bit. $0_B$ No action $1_B$ Set the flag and trigger an interrupt
<b>Product_Specific</b>	19, 25	rX	<b>Product_Specific</b> This bitfield is product specific.
<b>CRCs0</b>	20	w	<b>CRC Error Flag Set 0</b> This is write only bit. Writing 0 has no effect. Writing 1 triggers an interrupt and sets the corresponding flag bit. $0_B$ No action $1_B$ Set the flag and trigger an interrupt
<b>CRCs1</b>	21	w	<b>CRC Error Flag Set 1</b> This is write only bit. Writing 0 has no effect. Writing 1 triggers an interrupt and sets the corresponding flag bit. $0_B$ No action $1_B$ Set the flag and trigger an interrupt
<b>CRCs2</b>	22	w	<b>CRC Error Flag Set 2</b> This is write only bit. Writing 0 has no effect. Writing 1 triggers an interrupt and sets the corresponding flag bit. $0_B$ No action $1_B$ Set the flag and trigger an interrupt
<b>CRCs3</b>	23	w	<b>CRC Error Flag Set 3</b> This is write only bit. Writing 0 has no effect. Writing 1 triggers an interrupt and sets the corresponding flag bit. $0_B$ No action $1_B$ Set the flag and trigger an interrupt
<b>R1ES</b>	24	w	<b>RAMP1 Error Flag Set 3</b> This is write only bit. Writing 0 has no effect. Writing 1 triggers an interrupt and sets the corresponding flag bit. $0_B$ No action $1_B$ Set the flag and trigger an interrupt
<b>R1SS</b>	26	w	<b>RAMP1 Start Set</b> This is write only bit. Writing 0 has no effect. Writing 1 triggers an interrupt and sets the corresponding flag bit. $0_B$ No action $1_B$ Set the flag and trigger an interrupt
<b>0</b>	15:0, 31:27	r	<b>Reserved</b> Read as 0; should be written with 0.

## Radar Interface (RIF)

**Table 1149 Reset Values of FLAGSET**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	

### Flags Clear Register

The Flags Clear Register contains clear bits for the flag bits from the INTCON register.

#### FLAGSCL

##### Flags Clear Register

(0034<sub>H</sub>)

Reset Value: [Table 1150](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>LOCKI C</b>	<b>REGC RCC</b>	<b>SMCC</b>	<b>0</b>	<b>R1SC</b>	<b>Produ ct_Sp ecific</b>	<b>R1EC</b>	<b>CRCC3</b>	<b>CRCC2</b>	<b>CRCC1</b>	<b>CRCC0</b>	<b>Produ ct_Sp ecific</b>	<b>REC</b>	<b>FWC</b>	<b>CALC</b>	
w	w	w	r	w	rX	w	w	w	w	w	rX	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
<b>CALC</b>	16	w	<b>Calibration End Flag Clear</b> This is write only bit. Writing 0 has no effect. Writing 1 clears the flag bit. Reads as 0. 0 <sub>B</sub> No action 1 <sub>B</sub> Clear the flag
<b>FWC</b>	17	w	<b>Frame Watchdog Flag Clear</b> This is write only bit. Writing 0 has no effect. Writing 1 clears the flag bit. Reads as 0. 0 <sub>B</sub> No action 1 <sub>B</sub> Clear the flag
<b>REC</b>	18	w	<b>Ramp End Clear</b> This is write only bit. Writing 0 has no effect. Writing 1 clears the flag bit. Reads as 0. 0 <sub>B</sub> No action 1 <sub>B</sub> Clear the flag
<b>Product_Spec ific</b>	19, 25	rX	<b>Product_Specific</b> This bitfield is product specific.

## Radar Interface (RIF)

Field	Bits	Type	Description
<b>CRCC0</b>	20	w	<b>CRC Error Flag Clear 0</b> This is write only bit. Writing 0 has no effect. Writing 1 clears the flag bit. Reads as 0. $0_B$ No action $1_B$ Clear the flag
<b>CRCC1</b>	21	w	<b>CRC Error Flag Clear 1</b> This is write only bit. Writing 0 has no effect. Writing 1 clears the flag bit. Reads as 0. $0_B$ No action $1_B$ Clear the flag
<b>CRCC2</b>	22	w	<b>CRC Error Flag Clear 2</b> This is write only bit. Writing 0 has no effect. Writing 1 clears the flag bit. Reads as 0. $0_B$ No action $1_B$ Clear the flag
<b>CRCC3</b>	23	w	<b>CRC Error Flag Clear 3</b> This is write only bit. Writing 0 has no effect. Writing 1 clears the flag bit. Reads as 0. $0_B$ No action $1_B$ Clear the flag
<b>R1EC</b>	24	w	<b>RAMP1 Error Flag Clear</b> This is write only bit. Writing 0 has no effect. Writing 1 clears the flag bit. Reads as 0. $0_B$ No action $1_B$ Clear the flag
<b>R1SC</b>	26	w	<b>RAMP1 Start Flag Clear</b> This is write only bit. Writing 0 has no effect. Writing 1 clears the flag bit. Reads as 0. $0_B$ No action $1_B$ Clear the flag
<b>SMCC</b>	29	w	<b>SMCF Alarm Flag Clear</b> This is write only bit. Writing 0 has no effect. Writing 1 clears the flag bit INTCON.SMCF, if the enable bits of the safety features SPUCRCEN, LOCKIEN and REGCRCEN contain valid configuration 01 or 10, otherwise no effect. Reads as 0. $0_B$ 0 No action $1_B$ 1 Clear the flag
<b>REGCRCC</b>	30	w	<b>REGCRC Alarm Flag Clear</b> This is write only bit. Writing 0 has no effect. Writing 1 clears the INTCON.CRCF bit. Reads as 0. $0_B$ 0 No action $1_B$ 1 Clear the flag

## Radar Interface (RIF)

Field	Bits	Type	Description				
<b>LOCKIC</b>	31	w	<p><b>RIF Internal Lockstep Alarm Flag Clear</b></p> <p>This is a write only bit. Writing 0 has no effect. Writing 1 clears the INTCON.LOCKIF bit. Reads as 0.</p> <table> <tr> <td><math>0_B</math></td><td>0 No action</td></tr> <tr> <td><math>1_B</math></td><td>1 Clear the flag</td></tr> </table>	$0_B$	0 No action	$1_B$	1 Clear the flag
$0_B$	0 No action						
$1_B$	1 Clear the flag						
<b>0</b>	15:0, 28:27	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>				

**Table 1150** Reset Values of **FLAGSCL**

<b>Reset Type</b>	<b>Reset Value</b>	<b>Note</b>
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	

## Frame Watchdog Register

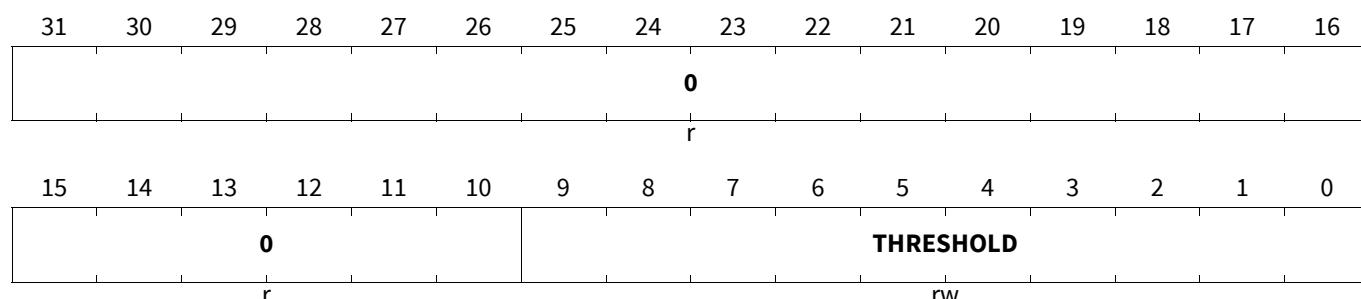
The FWDG register contains frame watchdog related bits.

FWDG

## Frame Watchdog Register

(0038<sub>H</sub>)

## **Reset Value: Table 1151**



Field	Bits	Type	Description
<b>THRESHOLD</b>	9:0	rw	<b>Frame Watchdog Threshold</b> Contains the reload value for the watchdog timer in the range of 0-1023. The counter counts with the kernel clock. In case of time-out, interrupt is raised and the sample counter is reset.
<b>0</b>	31:10	r	<b>Reserved</b> Read as 0; should be written with 0.

## Radar Interface (RIF)

**Table 1151 Reset Values of FWDG**

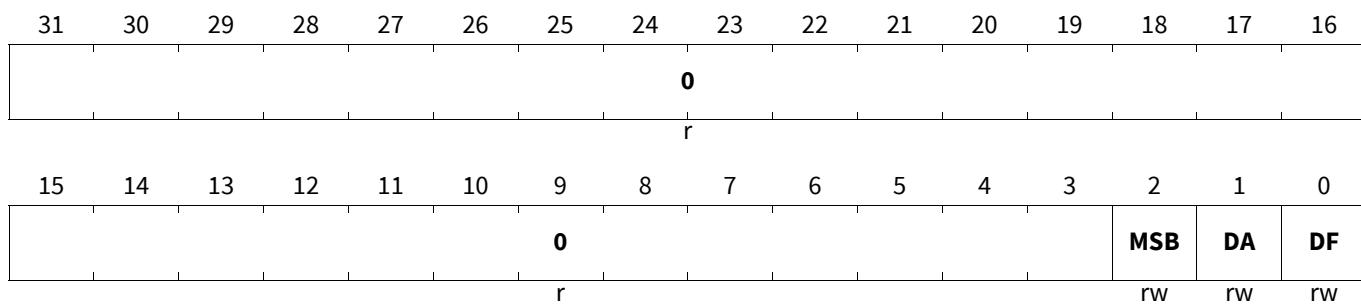
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	

### Data Formatting Unit Register

The DFU register contains configuration bits for the data formating options provided by the RIF module.

#### DFU

**Data Formatting Unit Register** (003C<sub>H</sub>) Reset Value: Table 1152



Field	Bits	Type	Description
DF	0	rw	<b>Data Format</b> Defines the type of data delivered to the Radar Interface by the ADC. 0 <sub>B</sub> Unsigned 1 <sub>B</sub> Signed
DA	1	rw	<b>Data Alignment</b> Defines the alignment of the data delivered to the SPU. 0 <sub>B</sub> Right (integer) 1 <sub>B</sub> Left (fractional)
MSB	2	rw	<b>Shift Direction MSB / LSB First</b> Defines the shift direction of the serial data, corresponding to the data bit on the lsb position in the delivered parallel data. 0 <sub>B</sub> MSB first 1 <sub>B</sub> LSB first
0	31:3	r	<b>Reserved</b> Read as 0; should be written with 0.

## Radar Interface (RIF)

**Table 1152 Reset Values of DFU**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	

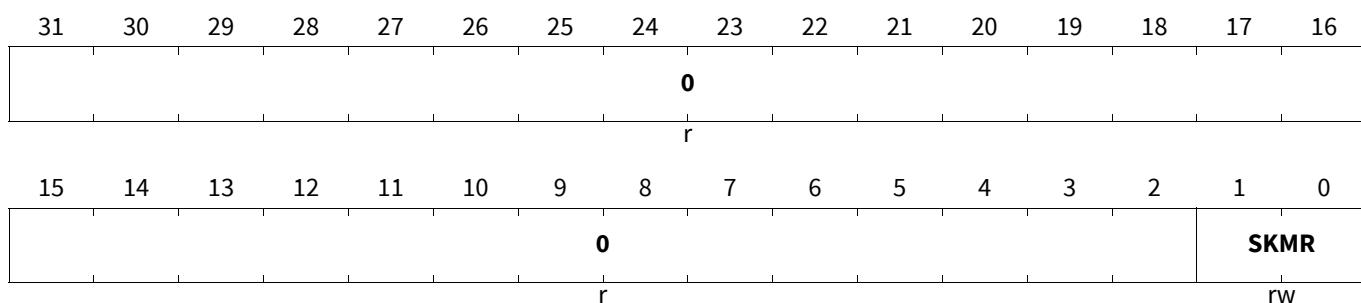
### SRIF Override Configuration Register

The SRIFOVRCFG register defines the granularity of the time measurement.

This register is not applicable for TC3Ax.

#### SRIFOVRCFG

**SRIF Override Configuration Register (0040<sub>H</sub>)** **Reset Value: Table 1153**



Field	Bits	Type	Description
<b>SKMR</b>	1:0	rw	<b>Skew Management Ratio</b> Ratio A of SKM Current Mirror. Defines the granularity of the time measurement. This field is for internal Infineon use and should not be modified from its default value. 00 <sub>B</sub> 3:1, default value, minimum accuracy 01 <sub>B</sub> 4:1 10 <sub>B</sub> 5:1, maximum accuracy for use in an application 11 <sub>B</sub> 6:1
<b>0</b>	31:2	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 1153 Reset Values of SRIFOVRCFG**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	

### Radar State Machine 2 Capture Register

The RSM2 register contains configuration parameters for the Radar State Machine.

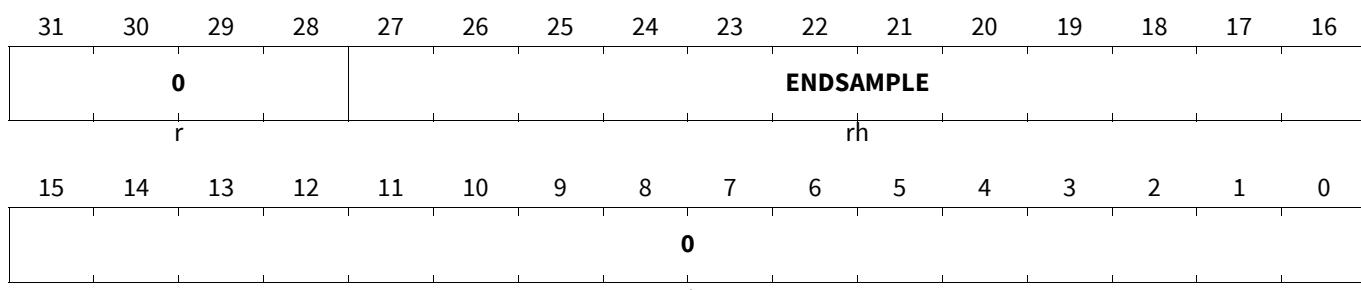
## Radar Interface (RIF)

## RSM2CAP

## Radar State Machine 2 Capture Register

(0044<sub>H</sub>)

Reset Value: Table 1154



Field	Bits	Type	Description
ENDSAMPLE	27:16	rh	<b>Value of the Current Sample at the End of the Ramp</b> This bitfield is updated by the value of the RSM1.CURSAMPLE upon End of Ramp event, before CURSAMPLE is reset to 0. Range of 0 to 2048 samples. ...      ... 000 <sub>H</sub> 0 001 <sub>H</sub> 1 002 <sub>H</sub> 2 800 <sub>H</sub> 2048 <b>others</b> , reserved
0	15:0, 31:28	r	<b>Reserved</b> Read as 0; should be written with 0.

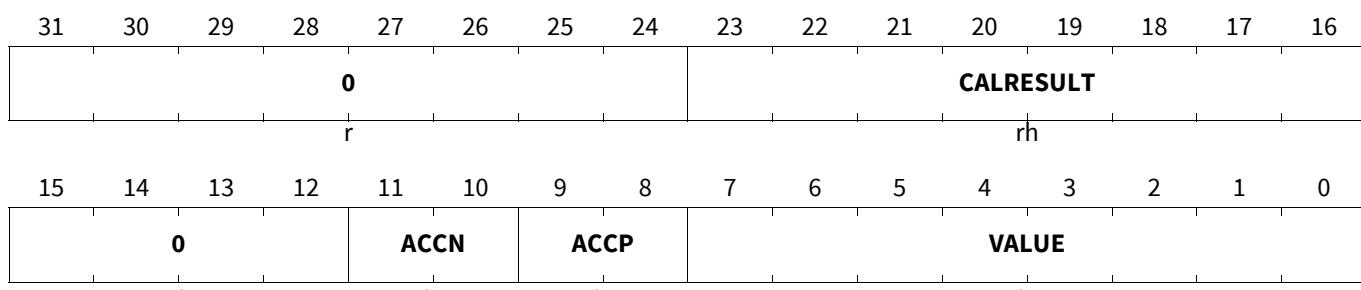
Table 1154 Reset Values of RSM2CAP

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	

## Skew Calibration Register

The SKEWCAL register contains the bit-fields related to the calibration of the delay lines of the SRIF block. This register is not applicable for TC3Ax.

## Radar Interface (RIF)

**SKEWCAL****Skew Calibration Register**(0048<sub>H</sub>)Reset Value: [Table 1155](#)

Field	Bits	Type	Description
<b>VALUE</b>	7:0	rw	<p><b>Calibration Word Configuring the Delay Lines</b></p> <p>The chip specific value for this field will be determined during production test. The reset value shall be overwritten by the tester determined value during initialisation and remains constant for the life time of the chip. This field is ENDINIT protected and must not be updated by application software.</p>
<b>ACCP</b>	9:8	rw	<p><b>Calibration Accuracy, Positive</b></p> <p>Contains the accuracy limits, relative to the target value defined in the VALUE bit field, to be used during the calibration process. The ACCP field defines the skew accuracy in the positive direction relative to the clock edge (later than the clock edge).</p> <p>The chip specific value for this field will be determined during production test. The reset value shall be overwritten by the tester determined value during initialisation and remains constant for the life time of the chip. This field is ENDINIT protected and must not be updated by application software.</p> <p> <math>00_B</math> +1  <math>01_B</math> +2  <math>10_B</math> +3  <math>11_B</math> +4     </p>
<b>ACCN</b>	11:10	rw	<p><b>Calibration Accuracy, Negative</b></p> <p>Contains the accuracy limits, relative to the target value defined in the VALUE bit field, to be used during the calibration process. The ACCN field defines the skew accuracy in the negative direction relative to the clock edge (earlier than the clock edge).</p> <p>The chip specific value for this field will be determined during production test. The reset value shall be overwritten by the tester determined value during initialisation and remains constant for the life time of the chip. This field is ENDINIT protected and must not be updated by application software.</p> <p> <math>00_B</math> -1  <math>01_B</math> -2  <math>10_B</math> -3  <math>11_B</math> -4     </p>

**Radar Interface (RIF)**

Field	Bits	Type	Description
<b>CALRESULT</b>	23:16	rh	<b>Calibration Word Resulting from Production Tester Calibration</b> This field is read during production test calibration to determine the reference skew to be programmed into the VALUE field.
<b>0</b>	15:12, 31:24	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 1155 Reset Values of SKEWCAL**

Reset Type	Reset Value	Note
Application Reset	0000 050F <sub>H</sub>	
CFS Value	00-- 050F <sub>H</sub>	SSW Register
Kernel Reset (software controlled by KRST0-1 registers)	0000 050F <sub>H</sub>	

**LVDS Control Register 0**

This register contains bits that control various properties of the LVDS pads. User Mode recommended value: 0606 0606<sub>H</sub>. Weak Driver Mode recommended value: 0707 0707<sub>H</sub>.

**LVDSCON0**

(004C <sub>H</sub> )																Reset Value: <a href="#">Table 1156</a>			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16				
DATA1																DATA0			
rw																rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	FRAME			
rw																rw			
CLK																			

## Radar Interface (RIF)

Field	Bits	Type	Description
<b>FRAME</b>	7:0	rw	<p><b>Frame LVDS Pad Control</b></p> <p>This field allows individual control of up to eight separate functions of the LVDS pad receiving the FRAME signal. Each of the functions can be enabled and disabled by setting and clearing the relevant bit of the register field as defined in the list below. Multiple functions can be enabled by setting multiple bits in the bitfield. Writing <math>00_H</math> to the bit field will disable all functions.</p> <ul style="list-style-type: none"> <li><math>01_H</math> Test Enable (enable Weak Driver Mode)</li> <li><math>02_H</math> Rterm Enable</li> <li><math>04_H</math> Frame Clock LVDS Pad Enable</li> <li><math>08_H</math> LVDS XOR-IN tied to LO by RIF-IP</li> <li><math>10_H</math> spare bit - tied to LO by RIF-IP</li> <li><math>20_H</math> LVDS pad enable T-Gate P Test</li> <li><math>40_H</math> LVDS pad enable T-Gate N-Test</li> <li><math>80_H</math> reserved</li> </ul>
<b>CLK</b>	15:8	rw	<p><b>CLOCK LVDS Pad Control</b></p> <p>This field allows individual control of up to eight separate functions of the LVDS pad receiving the CLK signal. Each of the functions can be enabled and disabled by setting and clearing the relevant bit of the register field as defined in the list below. Multiple functions can be enabled by setting multiple bits in the bitfield. Writing <math>00_H</math> to the bit field will disable all functions.</p> <ul style="list-style-type: none"> <li><math>01_H</math> Test Enable (enable Weak Driver Mode)</li> <li><math>02_H</math> Rterm Enable</li> <li><math>04_H</math> Frame Clock LVDS Pad Enable</li> <li><math>08_H</math> LVDS XOR-IN tied to LO by RIF-IP</li> <li><math>10_H</math> spare bit - tied to LO by RIF-IP</li> <li><math>20_H</math> LVDS pad enable T-Gate P Test</li> <li><math>40_H</math> LVDS pad enable T-Gate N-Test</li> <li><math>80_H</math> reserved</li> </ul>
<b>DATA0</b>	23:16	rw	<p><b>DATA0 LVDS Pad Control</b></p> <p>This field allows individual control of up to eight separate functions of the LVDS pad receiving the DATA0 signal. Each of the functions can be enabled and disabled by setting and clearing the relevant bit of the register field as defined in the list below. Multiple functions can be enabled by setting multiple bits in the bitfield. Writing <math>00_H</math> to the bit field will disable all functions.</p> <ul style="list-style-type: none"> <li><math>01_H</math> Test Enable (enable Weak Driver Mode)</li> <li><math>02_H</math> Rterm Enable</li> <li><math>04_H</math> Frame Clock LVDS Pad Enable</li> <li><math>08_H</math> LVDS XOR-IN tied to LO by RIF-IP</li> <li><math>10_H</math> spare bit - tied to LO by RIF-IP</li> <li><math>20_H</math> LVDS pad enable T-Gate P Test</li> <li><math>40_H</math> LVDS pad enable T-Gate N-Test</li> <li><math>80_H</math> reserved</li> </ul>

## Radar Interface (RIF)

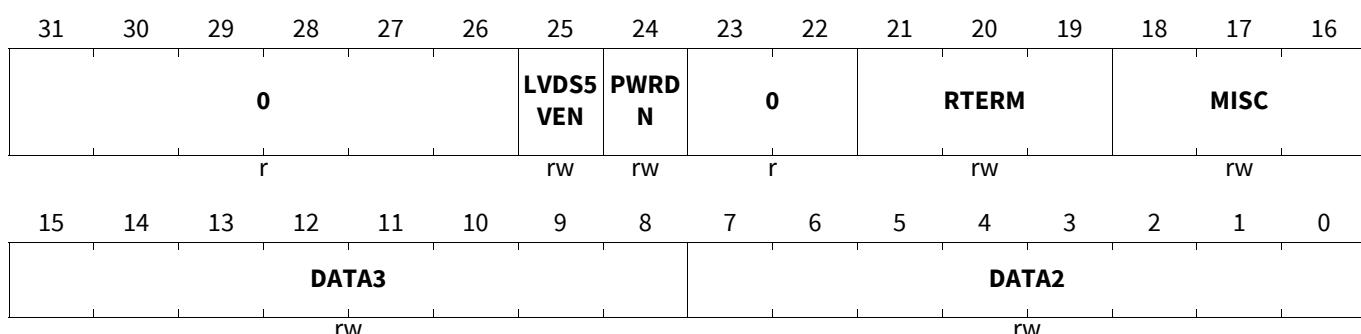
Field	Bits	Type	Description
<b>DATA1</b>	31:24	rw	<p><b>DATA1 LVDS Pad Control</b></p> <p>This field allows individual control of up to eight separate functions of the LVDS pad receiving the DATA1 signal. Each of the functions can be enabled and disabled by setting and clearing the relevant bit of the register field as defined in the list below. Multiple functions can be enabled by setting multiple bits in the bitfield. Writing <math>00_H</math> to the bit field will disable all functions.</p> <ul style="list-style-type: none"> <li><math>01_H</math> Test Enable (enable Weak Driver Mode)</li> <li><math>02_H</math> Rterm Enable</li> <li><math>04_H</math> Frame Clock LVDS Pad Enable</li> <li><math>08_H</math> LVDS XOR-IN tied to LO by RIF-IP</li> <li><math>10_H</math> spare bit - tied to LO by RIF-IP</li> <li><math>20_H</math> LVDS pad enable T-Gate P Test</li> <li><math>40_H</math> LVDS pad enable T-Gate N-Test</li> <li><math>80_H</math> reserved</li> </ul>

**Table 1156 Reset Values of LVDSCON0**

Reset Type	Reset Value	Note
Application Reset	$0000\ 0000_H$	
Kernel Reset (software controlled by KRST0-1 registers)	$0000\ 0000_H$	

**LVDS Control Register 1**

This register contains bits that control various properties of the LVDS pads. User Mode recommended value:  $0218\ 0606_H$ . Weak Driver Mode recommended value:  $0218\ 0707_H$ .

**LVDSCON1****LVDS Control Register 1****(0050<sub>H</sub>)****Reset Value: Table 1157**

## Radar Interface (RIF)

Field	Bits	Type	Description
<b>DATA2</b>	7:0	rw	<p><b>DATA2 LVDS Pad Control</b></p> <p>This field allows individual control of up to eight separate functions of the LVDS pad receiving the DATA2 signal. Each of the functions can be enabled and disabled by setting and clearing the relevant bit of the register field as defined in the list below. Multiple functions can be enabled by setting multiple bits in the bitfield. Writing <math>00_H</math> to the bit field will disable all functions.</p> <ul style="list-style-type: none"> <li><math>01_H</math> Test Enable (enable Weak Driver Mode)</li> <li><math>02_H</math> Rterm Enable</li> <li><math>04_H</math> Frame Clock LVDS Pad Enable</li> <li><math>08_H</math> LVDS XOR-IN tied to LO by RIF-IP</li> <li><math>10_H</math> spare bit - tied to LO by RIF-IP</li> <li><math>20_H</math> LVDS pad enable T-Gate P Test</li> <li><math>40_H</math> LVDS pad enable T-Gate N-Test</li> <li><math>80_H</math> reserved</li> </ul>
<b>DATA3</b>	15:8	rw	<p><b>DATA3 LVDS Pad Control</b></p> <p>This field allows individual control of up to eight separate functions of the LVDS pad receiving the FRAME signal. Each of the functions can be enabled and disabled by setting and clearing the relevant bit of the register field as defined in the list below. Multiple functions can be enabled by setting multiple bits in the bitfield. Writing <math>00_H</math> to the bit field will disable all functions.</p> <ul style="list-style-type: none"> <li><math>01_H</math> Test Enable (enable Weak Driver Mode)</li> <li><math>02_H</math> Rterm Enable</li> <li><math>04_H</math> Frame Clock LVDS Pad Enable</li> <li><math>08_H</math> LVDS XOR-IN tied to LO by RIF-IP</li> <li><math>10_H</math> spare bit - tied to LO by RIF-IP</li> <li><math>20_H</math> LVDS pad enable T-Gate P Test</li> <li><math>40_H</math> LVDS pad enable T-Gate N-Test</li> <li><math>80_H</math> reserved</li> </ul>
<b>MISC</b>	18:16	rw	<p><b>Miscellaneous Common LVDS Pad Control</b></p> <p>Controls some properties of all LVDS pads.</p> <ul style="list-style-type: none"> <li><math>000_B</math> disable 5V modes and low speed mode for all LVDS pads</li> <li><math>001_B</math> enable 5V Mode for all LVDS pads</li> <li><math>010_B</math> enable low speed mode for all LVDS pads</li> <li><math>100_B</math> reserved</li> </ul>
<b>RTERM</b>	21:19	rw	<p><b>Termination Resistor Trimming</b></p> <p>Trims the value of all termination resistors. The default value <math>100_B</math> should be updated by the application software using the appropriate value from the UCB_USER table. For RIF0, this is bits[2:0] of the RIF_LVDSCON1 entry and for RIF1, bits [18:16] of the same entry</p>
<b>PWRDN</b>	24	rw	<p><b>LVDS Bias Distributor Power Down</b></p> <p>Setting this bit powers down the LVDS bias distributor.</p> <ul style="list-style-type: none"> <li><math>0_B</math> active</li> <li><math>1_B</math> powered down</li> </ul>

## Radar Interface (RIF)

Field	Bits	Type	Description
<b>LVDS5VEN</b>	25	rw	<b>Enable 5V Mode for LVDS Bias Distributor</b> Setting this bit enables the 5V Mode for the LVDS Bias Distributor. $0_B$ disabled $1_B$ enabled
<b>0</b>	23:22, 31:26	r	<b>Reserved</b> Read as 0; should be written with 0.

**Table 1157 Reset Values of LVDSCON1**

Reset Type	Reset Value	Note
Application Reset	0321 0000 <sub>H</sub>	
CFS Value	-----00 0----- ----- <sub>B</sub>	The RTERM values for RIF0 and RIF1 are stored in the UCB_USER table in the RIF_LVDSCON1 entry. The RIF0 value is stored in the bits [2:0] of the entry, the RIF1 value in the bits [18:16] of the same entry. The application software has to copy these values from the UCB_USER table, entry RIF_LVDSCON1, to the LVDSCON1.RTERM bit field of each module correspondingly.
Kernel Reset (software controlled by KRST0-1 registers)	0321 0000 <sub>H</sub>	

## Debug Delay Register 0

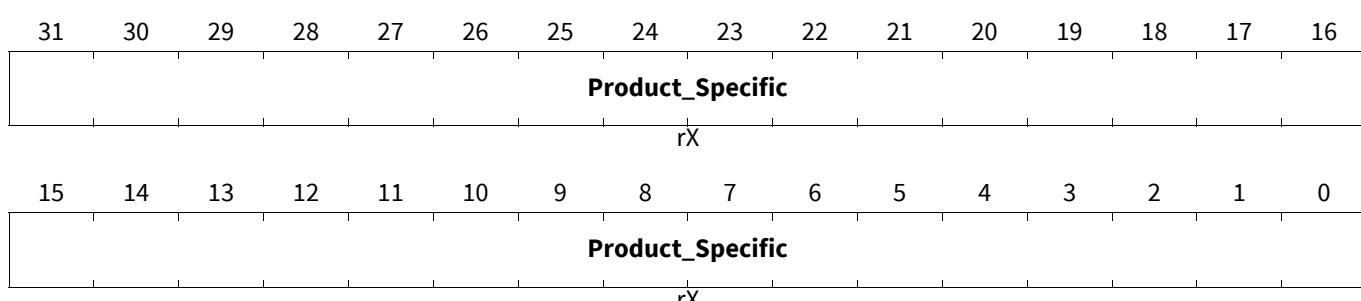
Read only register that displays the configuration values of the delay lines.

### DBGDLY0

#### Debug Delay Register 0

(0054<sub>H</sub>)

Reset Value: Table 1158



Field	Bits	Type	Description
<b>Product_Specifc</b>	31:0	rX	<b>Product_Specific</b> This bitfield is product specific.

## Radar Interface (RIF)

**Table 1158 Reset Values of **DBGDLY0****

Reset Type	Reset Value	Note
Application Reset	0033 3333 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0033 3333 <sub>H</sub>	

### Debug Delay Register 1

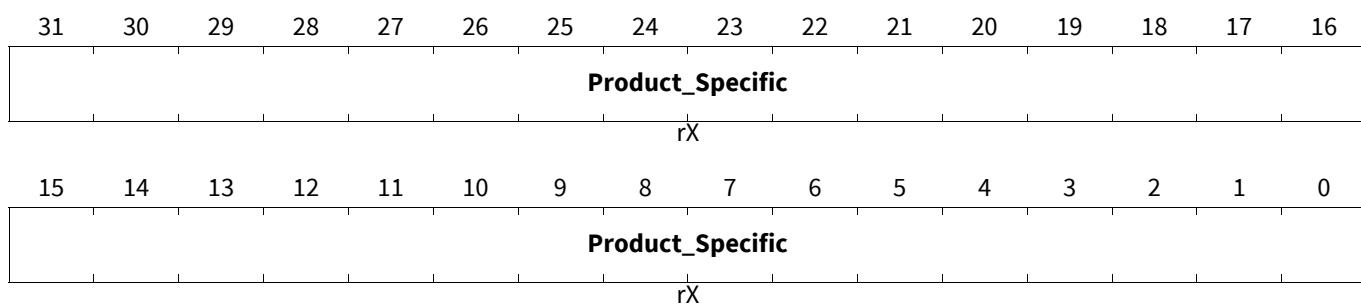
Read only register that displays the configuration values of the delay lines.

#### DBGDLY1

##### Debug Delay Register 1

(0058<sub>H</sub>)

Reset Value: [Table 1159](#)



Field	Bits	Type	Description
Product_Specific	31:0	rX	<b>Product_Specific</b> This bitfield is product specific.

**Table 1159 Reset Values of **DBGDLY1****

Reset Type	Reset Value	Note
Application Reset	3333 3333 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	3333 3333 <sub>H</sub>	

### DLL Control Register 0

This register is used to control and monitor DLL.

This is a product specific register and it is only applicable for TC3Ax.

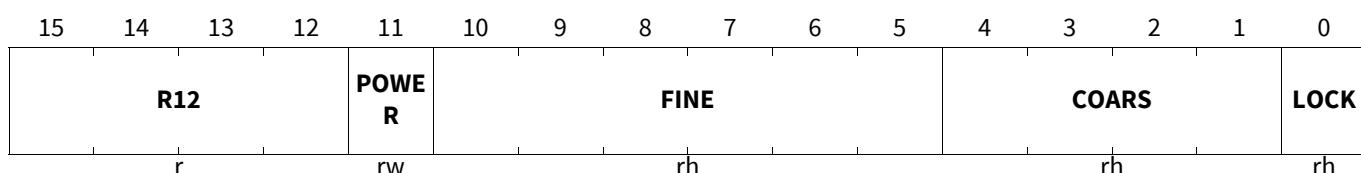
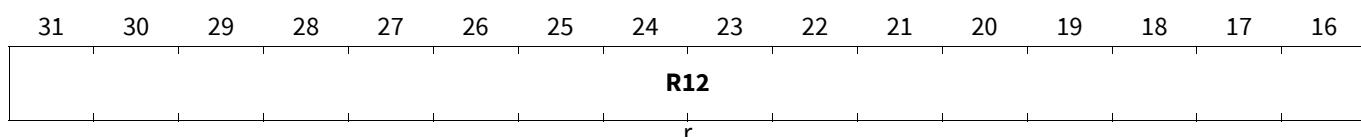
## Radar Interface (RIF)

### DLLCTL0

#### DLL Control Register 0

(005C<sub>H</sub>)

Reset Value: Table 1160



Field	Bits	Type	Description
LOCK	0	rh	<b>DLL lock status</b> Displays DLL lock status.
COARS	4:1	rh	<b>DLL coarse tuning value</b> Displays DLL coarse tuning value.
FINE	10:5	rh	<b>DLL fine tuning value</b> Displays DLL fine tuning value.
POWER	11	rw	<b>DLL power on</b> This bitfield is used to Power ON/OFF DLL.
R12	31:12	r	<b>Reserved - 0</b> Read as 0; should be written with 0.

**Table 1160 Reset Values of DLLCTL0**

Reset Type	Reset Value	Note
Application Reset	3333 3333 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	3333 3333 <sub>H</sub>	

### Debug Data Register 0

The DBG0 and DBG1 registers contain read-only debug information. The RIF captures the four first data samples received on a selected lane after module reset and makes the information available in the DD1, DD2, DD3 and DD4 register fields. The lane is selected by writing to the IPI.DBGSEL bitfield.

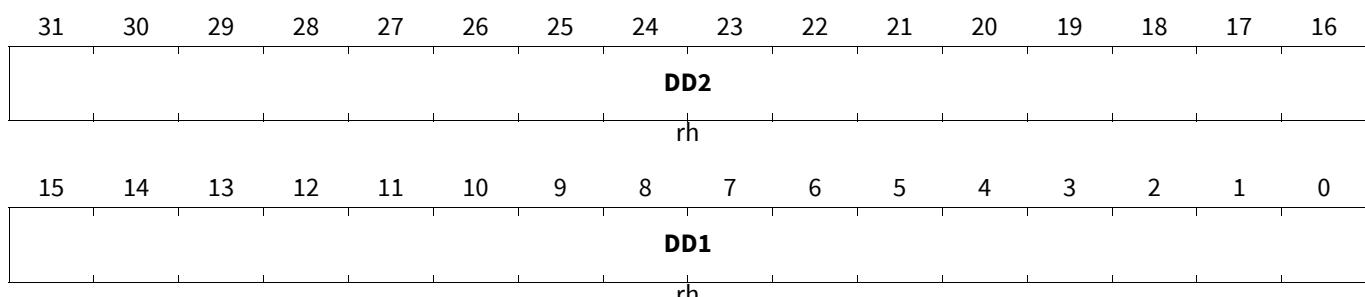
## Radar Interface (RIF)

### DBG0

#### Debug Data Register 0

(0080<sub>H</sub>)

Reset Value: Table 1161



Field	Bits	Type	Description
DD1	15:0	rh	<b>Debug Data First Sample</b> First 16 bit sample received from selected lane after module reset.
DD2	31:16	rh	<b>Debug Data Second Sample</b> Second 16 bit sample received from selected lane after module reset.

Table 1161 Reset Values of **DBG0**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	

### Debug Data Register 1

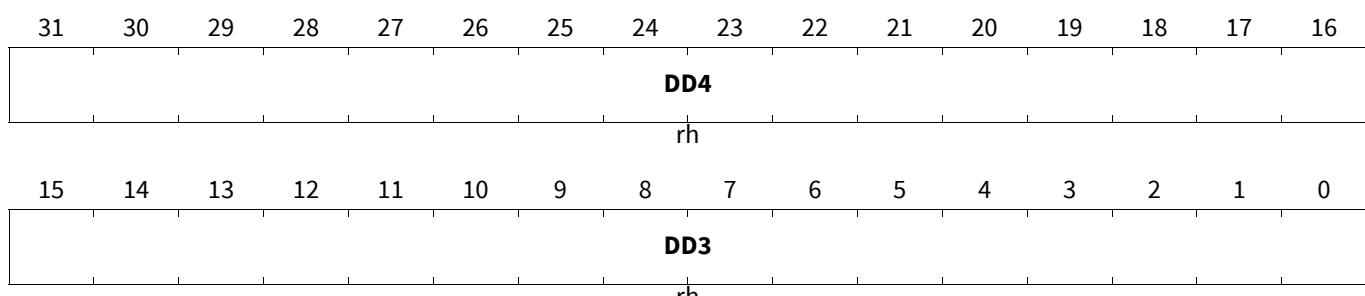
The DBG0 and DBG1 registers contain read-only debug information. The RIF captures the four first data samples received on a selected lane after module reset and makes the information available in the DD1, DD2, DD3 and DD4 register fields. The lane is selected by writing to the IPI.DBGSEL bitfield

### DBG1

#### Debug Data Register 1

(0084<sub>H</sub>)

Reset Value: Table 1162



Field	Bits	Type	Description
DD3	15:0	rh	<b>Debug Data Third Sample</b> Third 16 bit sample received from selected lane after module reset.

## Radar Interface (RIF)

Field	Bits	Type	Description
<b>DD4</b>	31:16	rh	<b>Debug Data Fourth Sample</b> Fourth 16 bit sample received from selected lane after module reset.

**Table 1162 Reset Values of DBG1**

Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	

## Safety Functions Register

The SFCON register contains control bits for the internal redundancy functions of the RIF module.

### SFCON

**Safety Functions Register** **Reset Value: Table 1163**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										SPUCRCEN		LOCKIEN			rw
r										rw		rw			

Field	Bits	Type	Description
<b>LOCKIEN</b>	1:0	rw	<b>Internal Lockstep Enable</b> Enables the redundant DFU block and the lockstep operation with the main DFU.  00 <sub>B</sub> invalid (and enabled) 01 <sub>B</sub> disabled (default) 10 <sub>B</sub> enabled 11 <sub>B</sub> invalid (and enabled)
<b>SPUCRCEN</b>	3:2	rw	<b>Enable Bit for the SPU CRC</b> Enables the SPU CRC generation.  00 <sub>B</sub> invalid (and disabled) 01 <sub>B</sub> disabled (default) 10 <sub>B</sub> enabled 11 <sub>B</sub> invalid (and enabled)
<b>0</b>	31:4	r	<b>Reserved</b> Read as 0; should be written with 0.

## Radar Interface (RIF)

**Table 1163 Reset Values of SFCON**

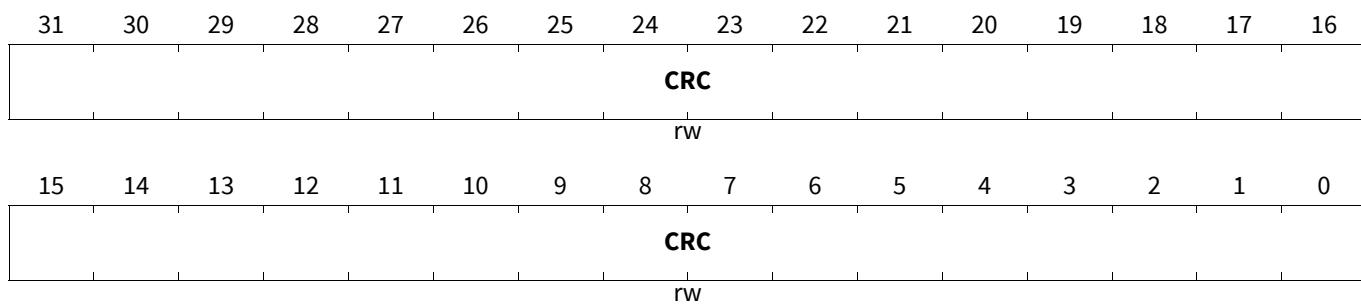
Reset Type	Reset Value	Note
Application Reset	0000 0005 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0005 <sub>H</sub>	

### Register CRC Register

The REGCRC register contains the pre-calculated expected value of the Register CRC safety feature.

#### REGCRC

**Register CRC Register** (008C<sub>H</sub>) **Reset Value: Table 1164**



Field	Bits	Type	Description
CRC	31:0	rw	<b>CRC Value</b> The pre-calculated expected value of the Register CRC.

**Table 1164 Reset Values of REGCRC**

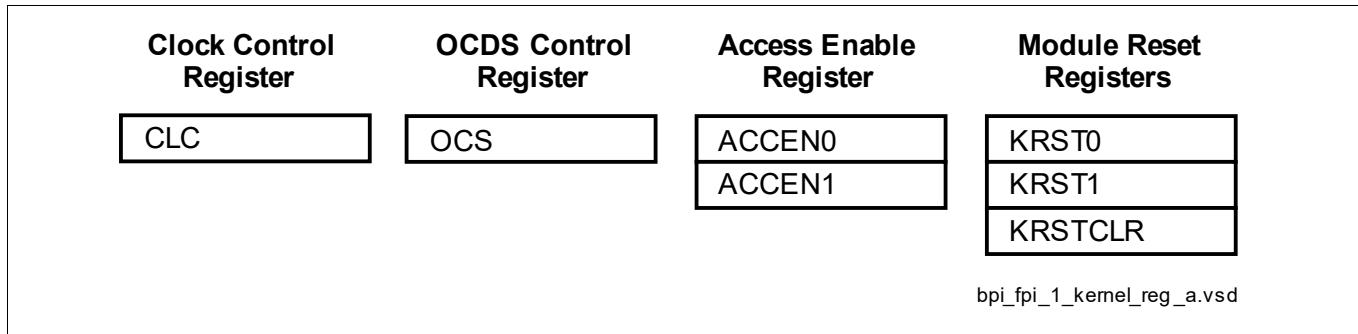
Reset Type	Reset Value	Note
Application Reset	0000 0000 <sub>H</sub>	
Kernel Reset (software controlled by KRST0-1 registers)	0000 0000 <sub>H</sub>	

### 24.4.2 BPI\_FPI Registers

Figure 325 shows all registers associated with the BPI\_FPI module, configured for one kernel.

## Radar Interface (RIF)

### BPI\_FPI Registers Overview



**Figure 325 BPI\_FPI Registers**

The writes of the bus masters to the RIF module are controlled by Access Protection registers ACCENx.

The RIF implements two ACCENx registers, ACCEN0 and ACCEN1.

The ACCENx registers are protected by Safety Endinit mechanism.

#### Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI\_FPI for the module. The CLC controls the  $f_{\text{ADAS}}$  module clock signal, sleep mode and disable mode for the module.

<b>CLC</b> <b>Clock Control Register</b>																<b>(0000<sub>H</sub>)</b>				<b>EEC Reset Value: 0000 0003<sub>H</sub></b>			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	0	r						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	0	r	rw	rw	rh	rw		

Field	Bits	Type	Description
<b>DISR</b>	0	rw	<b>Module Disable Request Bit</b> 0 <sub>B</sub> Enable, Request that the RIF clock tree be switched on 1 <sub>B</sub> Disable, Request that the RIF clock tree be switched off
<b>DISS</b>	1	rh	<b>Module Disable Status Bit</b> This bit will be set to 1 if the RIF kernel clock is disabled 0 <sub>B</sub> Enabled, The RIF clock tree is switched on 1 <sub>B</sub> Disabled, The RIF clock tree is switched off
<b>FDIS</b>	2	rw	<b>Freeze Disable</b> This bit controls the freeze function for this module. The freeze function is not implemented for the RIF so this bit will have no effect. This bit can be set to 0 <sub>B</sub>

## Radar Interface (RIF)

Field	Bits	Type	Description
<b>EDIS</b>	3	rw	<b>Sleep Mode Enable Control</b> Reserved. This bit currently has no effect on the RIF. It should be kept at 0 for future compatibility. 0 <sub>B</sub> Disabled, Sleep Mode is Off 1 <sub>B</sub> Enabled, Sleep Mode is on (no effect)
<b>0</b>	31:4	r	<b>Reserved</b> Read as 0; should be written with 0.

### OCDS Control and Status

The OCDS Control and Status (OCS) register is cleared by Debug Reset.

The register can only be written when the OCDS is enabled. If OCDS is being disabled, the OCS value will not change. When OCDS is disabled the OCS suspend control is ineffective. Write access is 32 bit wide only and requires Supervisor Mode.

### OCS

#### OCDS Control and Status (00E8<sub>H</sub>) Reset Value: Table 1165

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SUSST A	SUS_P		SUS							0				
r	rh	w		rw							r				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					0							TG_P	TGB	TGS	
					r							w	rw	rw	

Field	Bits	Type	Description
<b>TGS</b>	1:0	rw	<b>Trigger Set for OTGB0/1</b> 00 <sub>B</sub> No Trigger Set Output 01 <sub>B</sub> Trigger Set 1 10 <sub>B</sub> Reserved 11 <sub>B</sub> Reserved
<b>TGB</b>	2	rw	<b>OTGB0/1 Bus Select</b> 0 <sub>B</sub> Trigger Set is output on OTGB0 1 <sub>B</sub> Trigger Set is output on OTGB1
<b>TG_P</b>	3	w	<b>TGS, TGB Write Protection</b> TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
<b>SUS</b>	27:24	rw	<b>OCDS Suspend Control</b> Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 <sub>H</sub> Will not suspend 1 <sub>H</sub> Hard suspend. Clock is switched off immediately. 2 <sub>H</sub> reserved (will not suspend) <b>others</b> , reserved

## Radar Interface (RIF)

Field	Bits	Type	Description
<b>SUS_P</b>	28	w	<b>SUS Write Protection</b> SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
<b>SUSSTA</b>	29	rh	<b>Suspend State</b> 0 <sub>B</sub> Module is not (yet) suspended 1 <sub>B</sub> Module is suspended
<b>0</b>	23:4, 31:30	r	<b>Reserved</b> Read as 0; must be written with 0. The bits [4:0] are rw (readable and writable).

**Table 1165 Reset Values of OCS**

Reset Type	Reset Value	Note
Debug Reset	0000 0000 <sub>H</sub>	

### Access Enable Register 0

The Access Enable Register 0 controls write access<sup>1)</sup>for transactions with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI\_FPI is prepared for a 6-bit TAG ID. The registers ACCENO / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCENO.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ..., EN31 -> TAG ID 011111B. The BPI\_FPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers.

### ACCENO

(00FC <sub>H</sub> )																EEC Reset Value: FFFF FFFF <sub>H</sub>					
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16						
<b>EN31</b>	<b>EN30</b>	<b>EN29</b>	<b>EN28</b>	<b>EN27</b>	<b>EN26</b>	<b>EN25</b>	<b>EN24</b>	<b>EN23</b>	<b>EN22</b>	<b>EN21</b>	<b>EN20</b>	<b>EN19</b>	<b>EN18</b>	<b>EN17</b>	<b>EN16</b>						
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
<b>EN15</b>	<b>EN14</b>	<b>EN13</b>	<b>EN12</b>	<b>EN11</b>	<b>EN10</b>	<b>EN9</b>	<b>EN8</b>	<b>EN7</b>	<b>EN6</b>	<b>EN5</b>	<b>EN4</b>	<b>EN3</b>	<b>EN2</b>	<b>EN1</b>	<b>EN0</b>						
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

Field	Bits	Type	Description
<b>ENy (y=0-31)</b>	y	rw	<b>Access Enable for Master TAG ID y</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID y 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

1)

## Radar Interface (RIF)

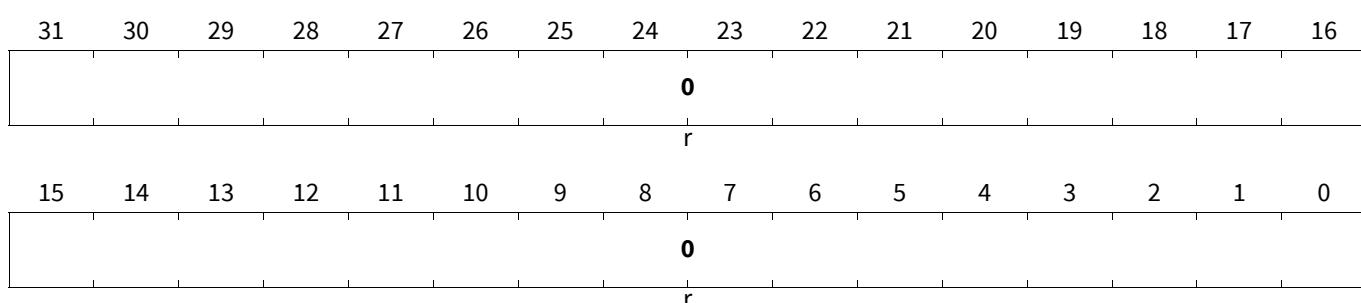
### Access Enable Register 1

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <> master peripheral mapping). The BPI\_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000B, EN1 -> TAG ID 100001B, ..., EN31 -> TAG ID 111111B.

#### ACCEN1

##### Access Enable Register 1 (00F8<sub>H</sub>) EEC Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
0	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

### Kernel Reset Register 0

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

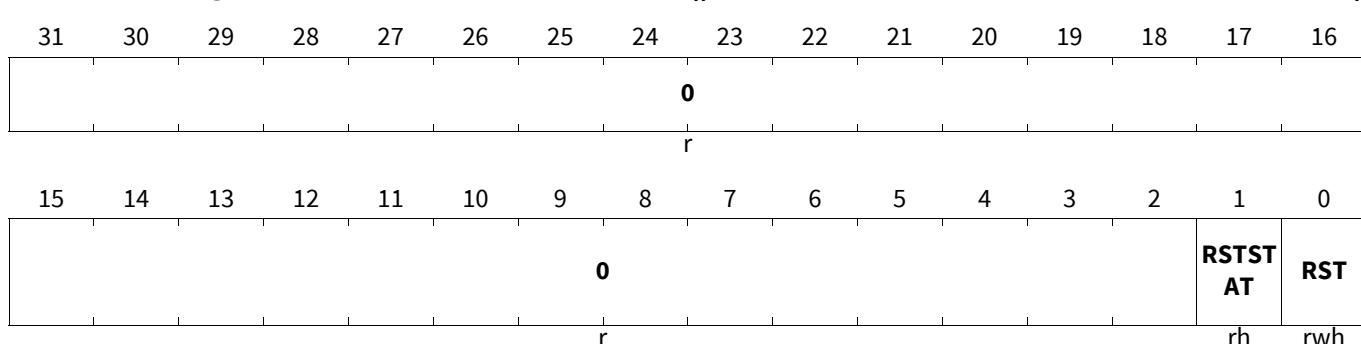
Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI\_FPI in the same clock cycle the RST bit is re-set by the BPI\_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLR.CLR register bit.

During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

The instance of the kernel reset interface registers, KRST0, KRST1 nad KRSTCLR, is an exception to the general rule that the kernel reset functionality protected by ENDINIT. This instance is protected only by the ACCEN protection mechanism.

#### KRST0

##### Kernel Reset Register 0 (00F4<sub>H</sub>) EEC Reset Value: 0000 0000<sub>H</sub>



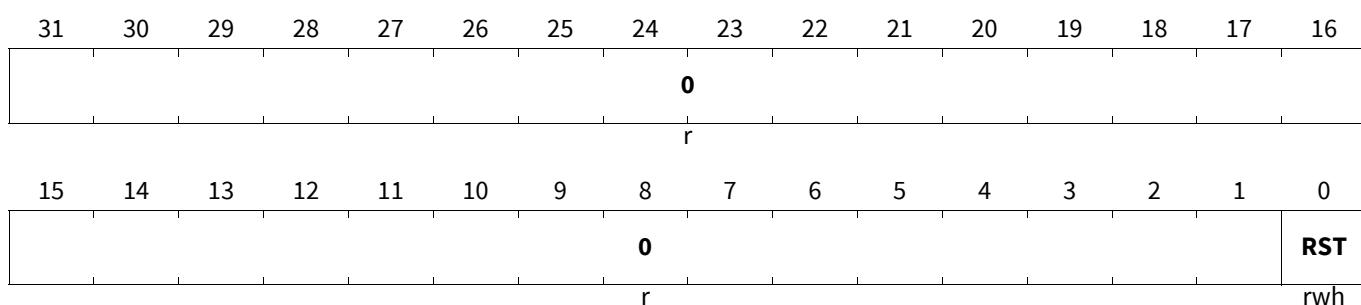
**Radar Interface (RIF)**

Field	Bits	Type	Description
<b>RST</b>	0	rwh	<p><b>Kernel Reset</b></p> <p>This bit can be used to request a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set.</p> <p>The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.</p> <p>0<sub>B</sub> No kernel reset was requested 1<sub>B</sub> A kernel reset was requested</p>
<b>RSTSTAT</b>	1	rh	<p><b>Kernel Reset Status</b></p> <p>This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits.</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.</p> <p>0<sub>B</sub> No kernel reset was executed 1<sub>B</sub> Kernel reset was executed</p>
<b>0</b>	31:2	r	<p><b>Reserved</b></p> <p>Read as 0; should be written with 0.</p>

**Kernel Reset Register 1**

The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

The instance of the kernel reset interface registers, KRST0, KRST1 nad KRSTCLR, is an exception to the general rule that the kernel reset functionality protected by ENDINIT. This instance is protected only by the ACCEN protection mechanism.

**KRST1****Kernel Reset Register 1**(00F0<sub>H</sub>)EEC Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>RST</b>	0	rwh	<p><b>Kernel Reset</b></p> <p>This bit can be used to request a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers are set.</p> <p>The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.</p> <p>0<sub>B</sub> No kernel reset was requested 1<sub>B</sub> A kernel reset was requested</p>

## Radar Interface (RIF)

Field	Bits	Type	Description
0	31:1	r	<b>Reserved</b> Read as 0; should be written with 0.

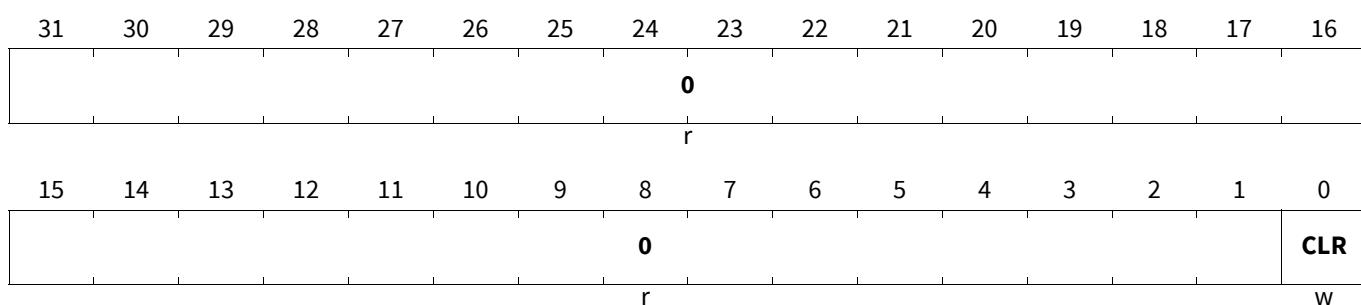
### Kernel Reset Status Clear Register

The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

The instance of the kernel reset interface registers, KRST0, KRST1 nad KRSTCLR, is an exception to the general rule that the kernel reset functionality protected by ENDINIT. This instance is protected only by the ACCEN protection mechanism.

#### KRSTCLR

**Kernel Reset Status Clear Register** (00EC<sub>H</sub>) EEC Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
CLR	0	w	<b>Kernel Reset Status Clear</b> Read always as 0. $0_B$ No action $1_B$ Clear Kernel Reset Status KRST0.RSTSTAT
0	31:1	r	<b>Reserved</b> Read as 0; should be written with 0.

## 24.5 IO Interfaces

**Table 1166 List of RIF Interface Signals**

Interface Signals	I/O	Description
ERR	out	<b>Radar Interface Service Request</b>
INT		
safety_alarm	out	<b>RIF Alarm</b>
RAMP1A	in	<b>External RAMP A input</b> External Ramp input port. This is an input from the external ramp generator in the Radar Front End indicating the 'valid' part of the ramp. It is considered as Asynchronous(?)
RAMP1B	in	<b>External RAMP B input</b>
RAMP1D	in	<b>External RAMP D input</b>

## Radar Interface (RIF)

**Table 1167 List of SRIF Interface Signals**

Interface Signals	I/O	Description
D1P	in	<b>LVDS RX Input (Data Bits of Channel #0)</b>
D1N	in	<b>LVDS RX Input (inverted Data Bits of Channel #0)</b>
D2P	in	<b>LVDS RX Input (Data Bits of Channel #1)</b>
D2N	in	<b>LVDS RX Input (inverted Data Bits of Channel #1)</b>
D3P	in	<b>LVDS RX Input (Data Bits of Channel #2)</b>
D3N	in	<b>LVDS RX Input (inverted Data Bits of Channel #2)</b>
D4P	in	<b>LVDS RX Input (Data Bits of Channel #3)</b>
D4N	in	<b>LVDS RX Input (inverted Data Bits of Channel #3)</b>
CLKP	in	<b>LVDS RX Input (Serial Clock)</b> CMOS signal from LVDS pad - Serial Clock - max. 200MHz
CLKN	in	<b>LVDS RX Input (inverted Serial Clock)</b> CMOS signal from LVDS pad - Serial Clock - max. 200MHz
FRP	in	<b>LVDS RX Input (FrameClock)</b> CMOS signal from LVDS pad - Frame Clock - max. 40MHz
FRN	in	<b>LVDS RX Input (inverted FrameClock)</b> CMOS signal from LVDS pad - Frame Clock - max. 40MHz

**24.6 Revision History****Table 1168 Revision History**

Reference	Change to Previous Version	Comment
V1.0.36		
<a href="#">Page 57</a>	Register field <b>SRIFOVRCFG</b> .SKMR marked as being for Infineon Internal use only.	
<a href="#">Page 62</a>	Description of register field, <b>LVDSCON1</b> .RTERM updated to be consistent with the reset table for the register	
<a href="#">Page 58</a>	Field Definitions for the <b>SKEWCAL</b> register updated	
<a href="#">Page 46</a>	Field Definitions for the <b>RSM1</b> register updated to clarify which reserved fields could potentially impact functionality	
<a href="#">Page 46</a>	Field Definitions for the <b>RSM1</b> register updated to clarify reserved fields for CHIRP signal support are not implemented.	
All	Sections specifically dealing with functions relating to the CHIRP signal input removed	
	<b>Section 24.1, Section 24.3.17.3,Section 24.3.17.2</b> Reference to “end of Chirp” event and CHIRP1 signal removed. Function shown as “unimplemented” where appropriate	
<a href="#">Page 48</a> , <a href="#">Page 51</a> , <a href="#">Page 53</a>	Field Definitions for the <b>INTCON</b> , <b>FLAGSSET</b> , <b>FLAGSCL</b> registers updated to redefine all fields related to CHIRP events as “unimplemented event”	
<a href="#">Page 41</a>	Description of <b>IPI</b> .PFP updated to explicitly define it as relevant for internal ADC operating mode only	

## Radar Interface (RIF)

**Table 1168 Revision History**

Reference	Change to Previous Version	Comment
<a href="#">Page 48</a>	Description of <b>INTCON</b> register updated to clarify that the status flags for event are only set if the relevant event also triggers an interrupt	
<a href="#">Page 25</a> , <a href="#">Page 25</a> , <a href="#">Page 27</a>	<b>Section 24.3.11.5 “Delay Adjustment During Calibration”</b> , <b>Section 24.3.11.6 “Skew Measurement During Calibration”</b> , <b>Section 24.3.11.7 “Reference Skew Value and Error Limits”</b> . New Sections added adding further detailed explanation of calibration.	
<a href="#">Page 24</a>	<b>Section 24.3.11.4 “Waveforms Required to Perform On-Chip Signal Delay Calibration”</b> expanded to provide more information on the calibration sequence	
<a href="#">Page 48</a>	Description of <b>INTCON</b> register updated to clarify the operation of the REF field	
<a href="#">Page 21</a>	<b>Section 24.3.11.1, “Frame Watchdog”</b> updated to clarify that no interrupts can be generated when the FWDG.THRESHOLD field is set to 0	
<a href="#">Page 19</a>	<b>Section 24.3.11, “External ADC Use-Case”</b> updated to better describe the two main options for the external ADC use case	
<b>V1.0.37</b>		
<a href="#">Page 48</a> , <a href="#">Page 51</a> , <a href="#">Page 53</a>	Field Definitions for the <b>INTCON</b> , <b>FLAGSSET</b> , <b>FLAGSCL</b> registers updated to remove name conflicts caused by duplicated mnemonics for UEE, UEF, UES & UEC bitfields	
<b>V1.0.38</b>		
<a href="#">Page 1</a>	New derivative TC3Ax	
<a href="#">Page 1</a> <a href="#">Page 23</a>	Feature list is updated to include TC3Ax features New sentence is added to <b>Chapter 24.3.11.2</b> to describe support for the new skew calibration procedure for TC3Ax	
<a href="#">Page 7</a>	Text added to <b>Section 24.3.4</b> to state that Ramp1 signal is not supported for TC3Ax	
<a href="#">Page 19</a>	Text added to <b>Section 24.3.11</b> to state that Ramp1 signal is not supported for TC3Ax	
<a href="#">Page 27</a>	Text added to <b>Section 24.3.11.8</b> to state that Ramp1 signal is not supported for TC3Ax	
<a href="#">Page 20</a> <a href="#">Page 35</a>	New <b>Figure 308</b> added for TC3Ax with no Ramp1 signal New <b>Figure 323</b> added for TC3Ax with no Ramp1 signal	
<a href="#">Page 13</a>	Paragraph added to <b>Section 24.3.8.1</b> to explain the new TC3Ax I/Q data support feature	
<a href="#">Page 16</a>	<b>Section 24.3.8.5</b> added to describe the new multiplex complex mode for TC3Ax	
<a href="#">Page 17</a>	<b>Figure 304</b> added to show the new multiplex complex mode for TC3Ax	
<a href="#">Page 6</a>	Clarification on avoiding the unintended Ramp1 error caused due to disabling deserializer	
<a href="#">Page 46</a> <a href="#">Page 47</a> <a href="#">Page 57</a> <a href="#">Page 21</a>	Clarified the conditions when CURRAMP is updated Clarified the conditions when CURSAMPLE is updated Clarified the conditions when ENDSAMPLE is updated Added details to clarify when the frame watchdog is started and stopped	

**Radar Interface (RIF)****Table 1168 Revision History**

<b>Reference</b>	<b>Change to Previous Version</b>	<b>Comment</b>
<a href="#">Page 72</a>	Register Descriptions are updated to clarify that they are not ENDINIT protected.	
<a href="#">Page 73</a>		
<a href="#">Page 74</a>		
<a href="#">Page 51</a>	For the following register bitfields “Unimplemented Events” are renamed into “Software Events” FLAGSSET[19]: UES -> SWE0S (Software Event 0 Flag Set) FLAGSSET[25]: UES -> SWE1S (Software Event 1 Flag Set)	
<a href="#">Page 51</a>	FLAGSCL[19]: RR19 -> SWE0C (Software Event 0 Flag Clear) FLAGSCL[25]: RR25 -> SWE1C (Software Event 1 Flag Clear)	
<a href="#">Page 53</a>	INTCON[3]: UEE -> SWE0E (Software Event 0 Interrupt Enable) INTCON[9]: UEE -> SWE1E (Software Event 1 Interrupt Enable)	
<a href="#">Page 48</a>	INTCON[19]: UEF -> SWE0F (Software Event 0 Interrupt Flag) INTCON[25]: UEF -> SWE1F (Software Event 1 Interrupt Flag)	
<a href="#">Page 43</a>	<b>FLM</b> Register bitfields are updated to support new TC3Ax features	
<a href="#">Page 46</a>	Following register bitfield descriptions are updated to indicate the discontinued Ramp1 and Chirp1 signal support for TC3Ax: <b>RSM1</b>	
<a href="#">Page 48</a>	<b>INTCON</b>	
<a href="#">Page 51</a>	<b>FLAGSSET</b>	
<a href="#">Page 53</a>	<b>FLAGSCL</b>	
<a href="#">Page 37</a>	In <a href="#">Table 1140</a> , access modes for SFCON are changed from SV, SE to SV, P	

**V1.0.39**

<a href="#">Page 69</a>	Bit 2 of <b>CLC</b> register is changed from reserved to FDIS to align with the RTL	
<a href="#">Page 16</a>	Clarification added regarding the use of Multiplex Complex Mode	
<a href="#">Page 17</a>	Figure updated for clarity	
<a href="#">Page 41</a>	Application reset value is added	
<a href="#">Page 43</a>	Application Reset value is updated	
	Product specific bitfields are shown in appendices	
<a href="#">Page 48</a>	Product specific bitfields are shown in appendices	
	RISF register bitfield description is updated	
<a href="#">Page 51</a>	Product specific bitfields are shown in appendices	
	RISS register bitfield description is updated	
<a href="#">Page 53</a>	RISC register bitfield description is updated	

**V1.0.40**

<a href="#">Page 16</a>	Clarification is added to indicate real and imaginary data components in <a href="#">Figure 304</a>	
<a href="#">Page 2</a>	<a href="#">Table 1137</a> is added to show RIF identification numbers for each derivative	
<a href="#">Page 13</a>	In <a href="#">Page 13</a> , paragraph explaining I-Q support for TC3Ax is updated	
<a href="#">Page 16</a>	In <a href="#">Section 24.3.8.5</a> , references to real and imaginary components is updated Text explaining lack of I-Q multiplexing support for internal ADC usecase is added <b>FLM FORMAT</b> is changed to <b>FLM.MULCMPX</b>	

## Radar Interface (RIF)

**Table 1168 Revision History**

Reference	Change to Previous Version	Comment
<a href="#">Page 70</a>	OEN write protection access mode is added for <b>OCS</b>	
<a href="#">Page 39</a>	<b>ID</b> register description us updated	
<a href="#">Page 39</a>	For registers from <b>ESI</b> to <b>REGCRC</b> , Kernel reset values are added for clarification	
<a href="#">Page 46</a>	Product specific bitfields are moved to appendices	
<a href="#">Page 62</a>	<b>LVDSCON1.MISC</b> bitfield is updated	
<a href="#">Page 58</a>	<b>SKEWCAL</b> register description is updated to indicate that it is redundant for TC3Ax	
<a href="#">Page 57</a>	<b>SRIFOVRCFG</b> register description is updated to indicate that it is redundant for TC3Ax	
<a href="#">Page 27</a>	A note is added to indicate that SKEWCAL register is not applicable for TC3Ax	
<a href="#">Page 64</a>	Product specific registers <b>DBGDLY0</b> and <b>DBGDLY1</b> are moved to appendices	
<a href="#">Page 48</a>	Device specific registers ( <b>INTCON</b> , <b>FLAGSSET</b> , <b>FLAGSCL</b> ) are moved to appendix	
<a href="#">Page 51</a>		
<a href="#">Page 53</a>		
<a href="#">Page 65</a>	<b>DLLCTL0</b> register is added which is device specific for TC3Ax	
<a href="#">Page 39</a>	Product specific register <b>ESI</b> is moved to appendix	

## High Speed Pulse Density Modulation Module (HSPDM)

### 25 High Speed Pulse Density Modulation Module (HSPDM)

The HSPDM is intended to generate up to two 1-bit bit-streams. Each bit-stream represents the 16-bit data stored inside the dedicated 8 KB SRAM. This bit-stream is a pulse-density modulated (PDM) bit-stream which can be averaged outside the microcontroller using a low pass filter (LPF) to generate the analog voltage. [Figure 326](#) shows the architecture of the HSPDM and its interface to other IPs. [Figure 326](#) also highlights the signal flow from SRAM to bit-stream blocks at the output pads. In a system, HSPDM is conceived to generate an analog signal in conjunction with a LPF implemented external to the microcontroller. The signal-to-noise ratio (SNR) of the generated analog voltage is a function of the cutoff frequency, order and the implementation of the LPF and the frequency of operation of the Bit-Stream block.

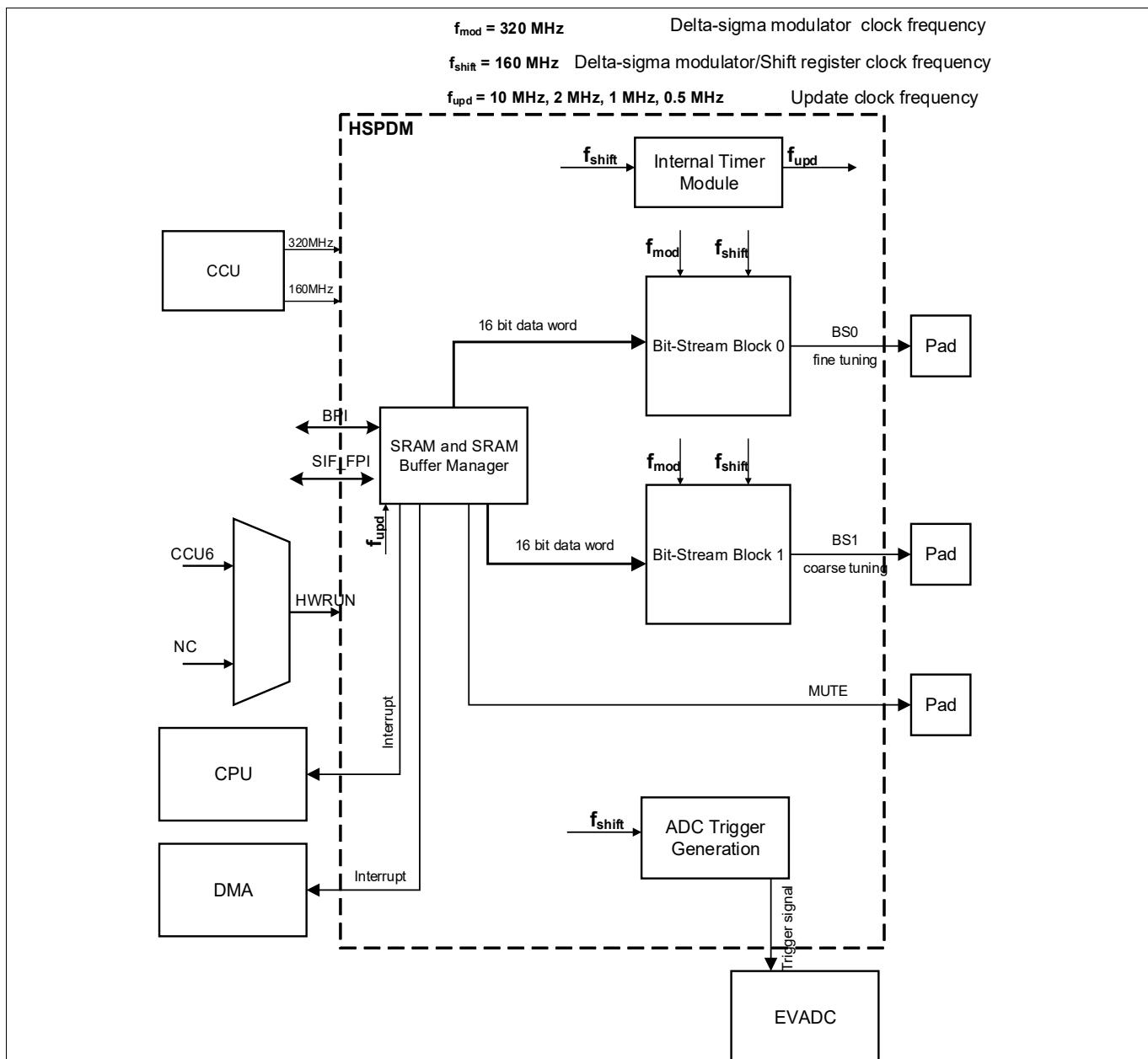


Figure 326 High speed pulse density modulation block level diagram

#### 25.1 Feature List

This section lists the features of the HSPDM module:

## High Speed Pulse Density Modulation Module (HSPDM)

- Two independent synchronous PDM bit-streams using the Delta-sigma modulator or bit-streams shifted serially using the shift register up to 160 Mbps
- Trigger signal to EVADC to signal the start of conversion
- Programmable offset in EVADC trigger signal
- MUTE signal output from the microcontroller which can be used to turn on or turn off the external transmitter for details see [Chapter 25.2.4.2](#)
- Support burst mode for fast upload into the SRAM through an addition slave interface (SIF)

## 25.2 Functional Description

### 25.2.1 HSPDM Modes of Operation

HSPDM generates up to two bit-streams with bit-stream block 0 (BSB0) and bit-stream block 1 (BSB1). The correct mode of operation must be set by the user using **CON.SM**. The user must not change the mode of operation during the run time. Before changing the mode of operation, the user must stop the bit-streaming by setting the **CON.RUNC** bit, followed by setting the correct mode of operation using the **CON.SM** bit and then restarting the bit-streaming by setting the **CON.RUNS** bit. Mode changes during the run time will result in an undefined behavior of the bit-stream. Each bit-stream block supports up to three modes of operation ([Figure 327](#)):

- Shift register generated bit-stream
- Delta-sigma modulator generated bit-stream with the CIC filter and the Compactor enabled
- Delta-sigma modulator generated bit-stream with the CIC filter and the Compactor disabled

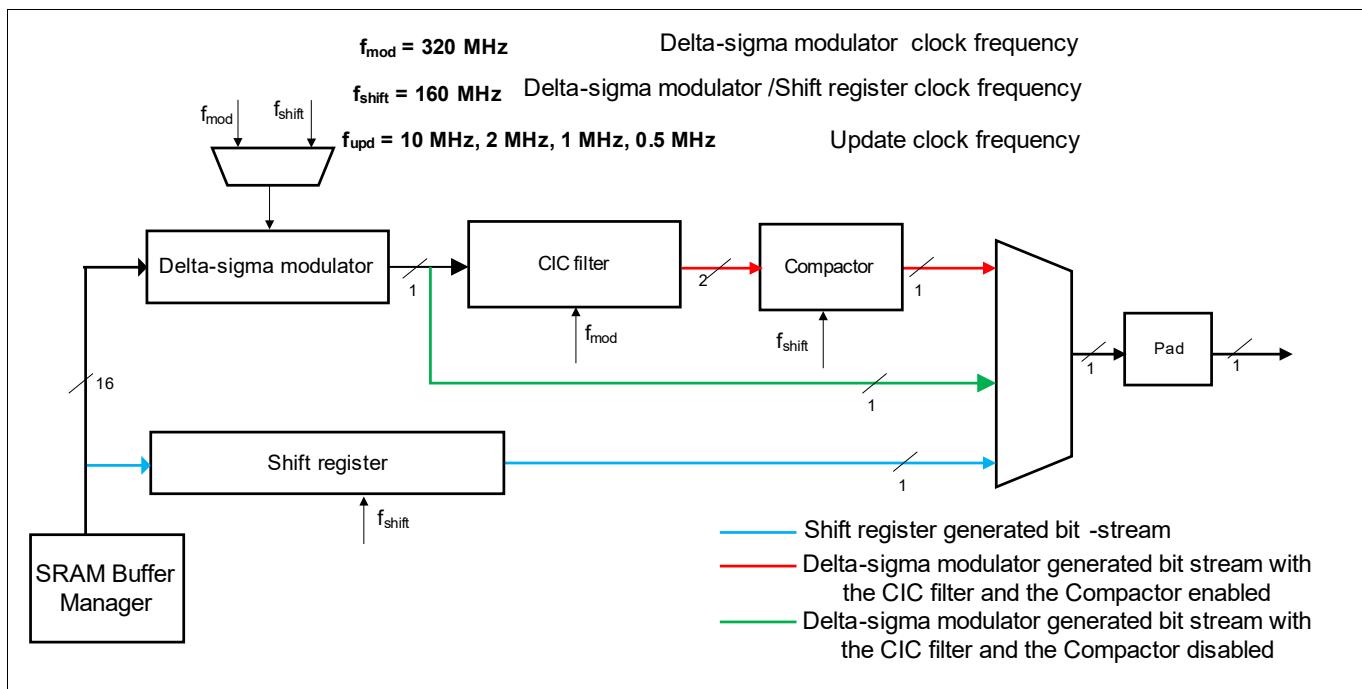


Figure 327 HSPDM modes of operation

#### 25.2.1.1 Shift Register Generated Bit-Stream

The mode can be entered by setting **CON.SM** = 10<sub>B</sub>. In this mode of operation a 16-bit digital word is loaded by the bit-stream loader (inside the SRAM & SRAM buffer manager block) at the input of the shift register. The Delta-sigma modulator, the CIC filter and the Compactor are disabled in this mode of operation. The shift register

## High Speed Pulse Density Modulation Module (HSPDM)

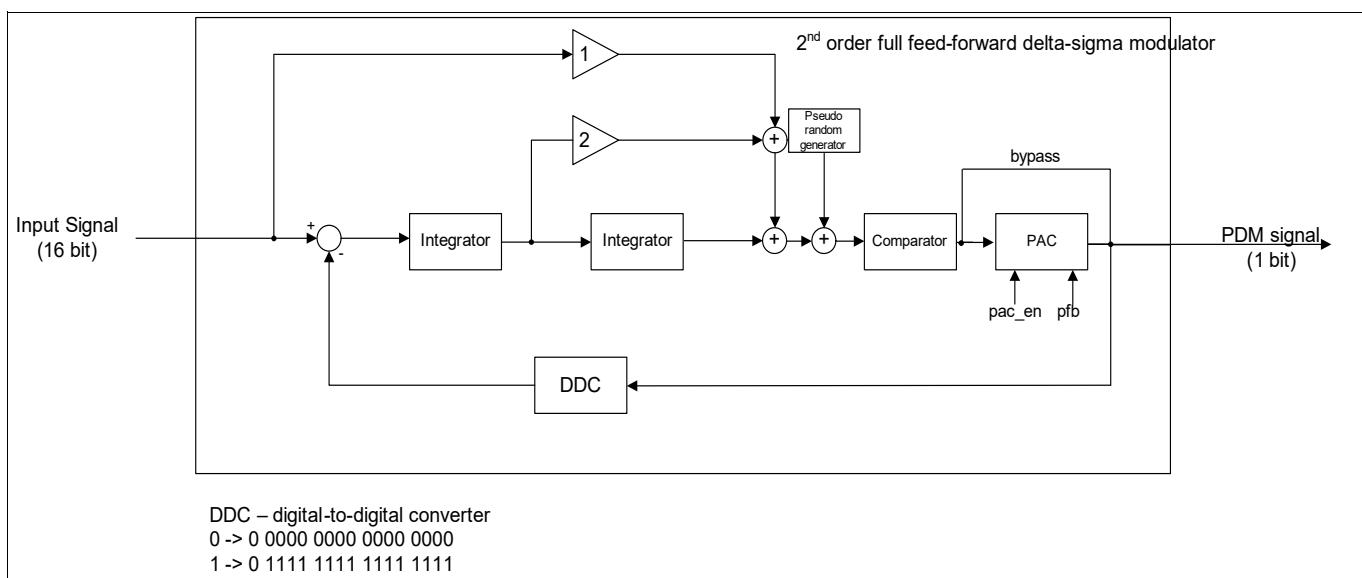
serializes the 16-bit word and sends it bit-wise (lsb first) on the rising edge of the  $f_{shift}$  clock. At the output, a 1-bit bitstream at 160 Mbps is generated which can be pushed out of the microcontroller through the multiplexer. The update frequency of the loader  $f_{upd}$  must be 10 MHz in this mode. User must take care to configure the correct  $f_{upd}$  during this mode using **CON.ITMDIV**.

**Note:** System SNR performance is limited during this mode.

### 25.2.1.2 Delta-sigma Modulator Generated Bit-Stream with the CIC filter and the Compactor enabled

The mode can be entered by setting **CON.SM** = 00<sub>B</sub>. In this mode of operation, a 16-bit digital word is loaded by the bit-stream loader (inside the SRAM & SRAM buffer manager block) at the input of the Delta-sigma modulator. Shift register is disabled in this mode of operation. The Delta-sigma modulator generates a PDM bit-stream at its output. The Delta-sigma modulator runs at 320 MHz in this mode, generating PDM bit-stream at 320 Mbps. The data rates higher than 160 Mbps are not supported by the microcontroller pads, therefore, in this mode of operation the output of the modulator is decimated.

In the HSPDM, a 2<sup>nd</sup> order full feed-forward Delta-sigma modulator (**Figure 328**) is used. The full feed-forward architecture offer the advantage of higher stability due to the fact that the inputs to the integrators are the difference between the input and the feedback signal. Therefore, the swings at the input of the integrators are much smaller than the full scale value. As depicted in the **Figure 328**, the feedback of the Delta-sigma modulator is a digital-to-digital converter (DDC) which converts a 1-bit output from the comparator to the 17-bit value to be subtracted from the input. The pulse density of the generated bit-stream depends on the absolute input level with respect to the full scale value. **Table 1169** demonstrates some PDM output bit-streams from the Delta-sigma modulator. To showcase the pulse density, **Table 1169** assumes a full scale value of  $2^{16}$ , in actual implementation the full scale value is  $2^{16}-1$ .



**Figure 328 Full feed forward Delta-sigma modulator**

**Table 1169 PDM Bit-Stream Examples**

Input to the Delta-sigma modulator	Output of the Delta-sigma modulator
0000 <sub>H</sub>	0000000000000000....
8000 <sub>H</sub>	1100110011001100.....

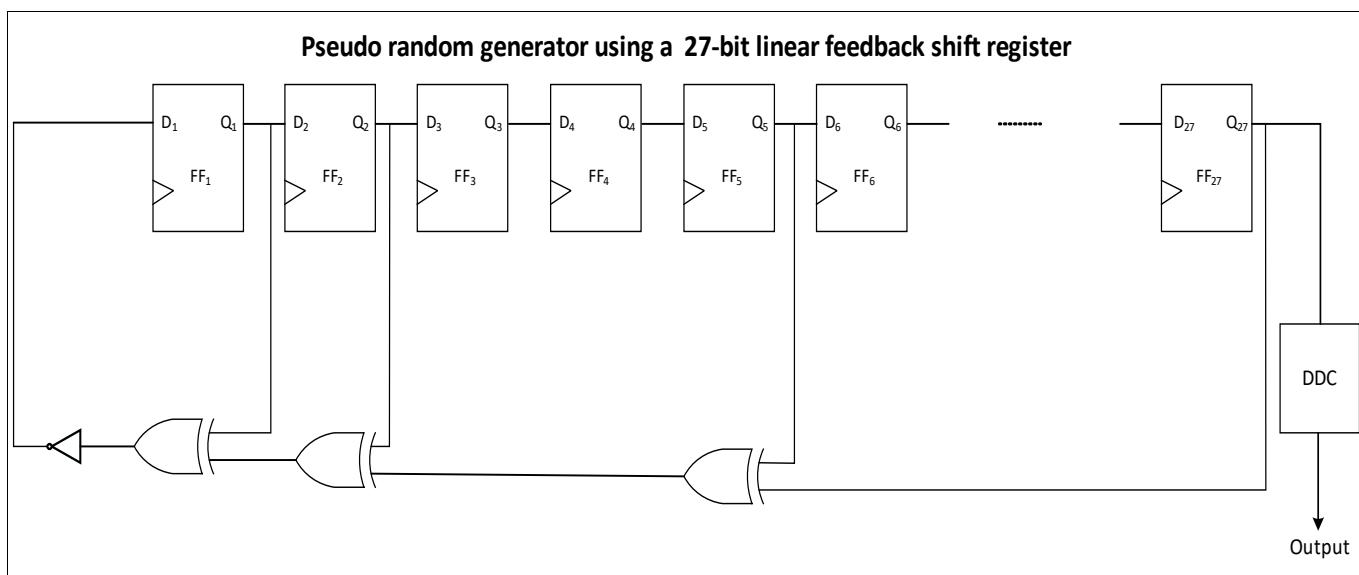
## High Speed Pulse Density Modulation Module (HSPDM)

**Table 1169 PDM Bit-Stream Examples**

Input to the Delta-sigma modulator	Output of the Delta-sigma modulator
4000 <sub>H</sub>	1000010010000100....
FFFF <sub>H</sub>	1111111111111111....

For DC or low frequency input signal the output of the modulator contains periodic quantization noise components and some of these noise components could fall within the passband of interest and limit the dynamic range of the modulator. This phenomenon is called ‘Limit cycling’. Limit cycling results in spurious tones in the output spectrum of the modulator and deteriorates the performance. To overcome the impact of limit cycle, white noise is added in after the second integrator inside of delta sigma modulator to dither the output and thereby, reduce or eliminate the level of the periodic quantization noise. Adding of noise increases the overall noise floor and reduces the effective signal-to-noise ratio, therefore, the level of dither must be chosen efficiently by the user.

Dither is implemented inside the modulator using a pseudo-random generator ([Figure 328](#)). Pseudo-random generator is implemented with the help of 27-bit Linear Feedback Shift Register (LFSR)



**Figure 329 Dither generation**

A 27-bit LFSR results in a period of dither of  $2^{27}-1$  samples or 839 ms for 160 MHz clock. The dither levels are defined in [Table 1170](#). User can configure the described dither levels using [CON.DITH](#).

**Table 1170 Dither Levels**

Setting	Dither Level
000	Disabled
001	Minimum dither level
010	Low dither level
011	Low-medium dither level
100	Medium dither level
101	Medium-high dither level
110	High dither level
111	Highest dither level

## High Speed Pulse Density Modulation Module (HSPDM)

Decimation in HSPDM is implemented in two steps using the CIC filter and the Compactor. The CIC filter (Figure 330) follows the Delta-sigma modulator and decimates the 1-bit PDM signal from the Delta-sigma modulator at 320 Mbps to 160 MSps, with each sample size of 2-bit. This 2-bit sample at the output of the CIC filter must be mapped to a 1-bit sample without any loss of information. This task is accomplished by the Compactor which maps the 2-bit sample into 1-bit.

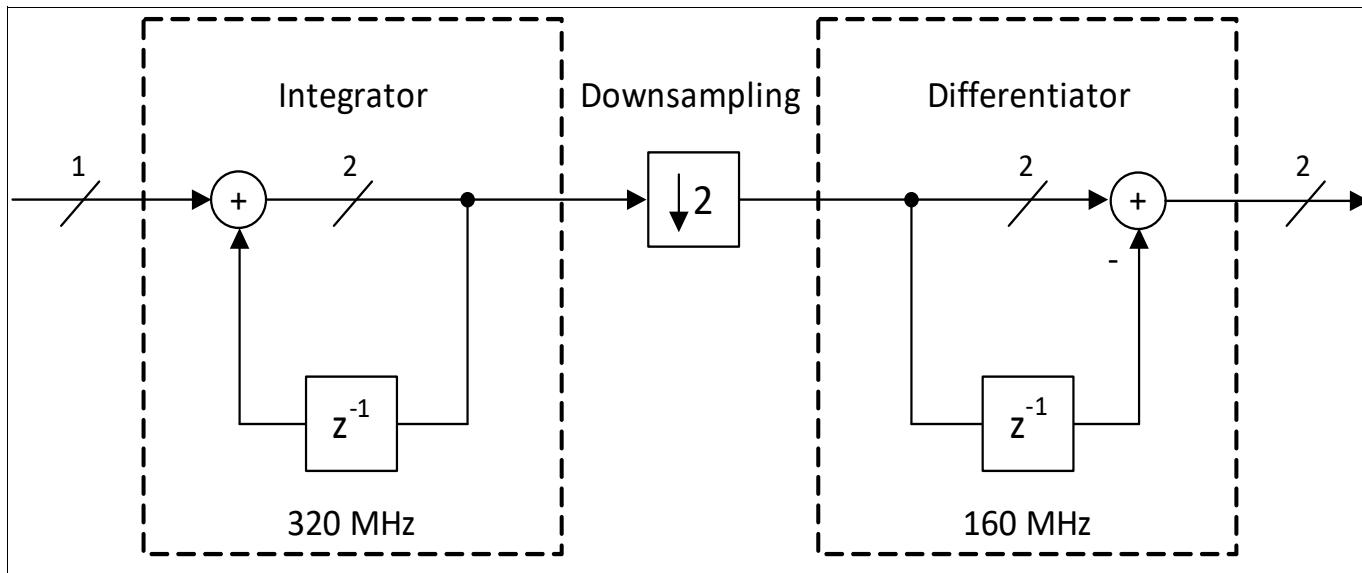


Figure 330 CIC filter with decimation by a factor of 2

Inside of the Compactor a mapping algorithm is implemented, in which a ' $0_D$ ' input is mapped to a ' $0_B$ ', a ' $2_D$ ' input is mapped to a ' $1_B$ ' and for ' $1_D$ ' as input, Compactor flips the output bit between ' $0_B$ ' and ' $1_B$ '. The algorithm is initialized: when for the very first time a ' $1_D$ ' comes at the input of the Compactor a ' $0_B$ ' is mapped at the output. After that whenever a ' $1_D$ ' is generated at the output of the CIC, the Compactor flips the output between a ' $0_B$ ' and a ' $1_B$ ' based on whether the previous output for input ' $1_D$ ' was ' $1_B$ ' or ' $0_B$ ', respectively. An example demonstrating the functionality of the mapping algorithm is shown in Figure 331 and Table 1171 explains the truth table for the mapping algortihm of the Compactor.

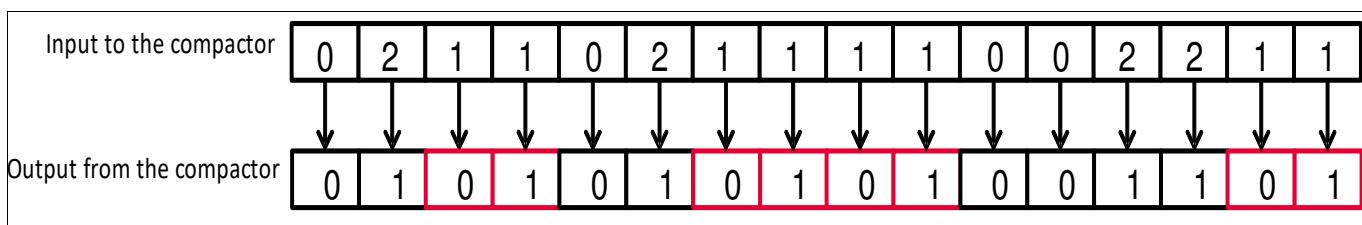


Figure 331 Mapping algorithm

Table 1171 Mapping algorithm of the Compactor

Input to the Compactor	Output of the Compactor
$0_D$	$0_B$
$1_D$	$0_B$ (at the start or if the previous output was 1) $1_B$ (if the previous output was 0)
$1_D$	$1_B$
$2_D$	$1_B$

## High Speed Pulse Density Modulation Module (HSPDM)

During this mode of operation, the default update frequency of the loader,  $f_{upd}$ , is 1 MHz. User can also program the internal timer module [Chapter 25.2.2.1 CON](#).ITMDIV to change the update frequency in this mode. A  $f_{upd}$  of 10 MHz is not allowed in this mode.

### 25.2.1.3 Delta-sigma Modulator Generated Bit-Stream with the CIC filter and the Compactor disabled

The mode can be entered by setting [CON.SM](#) = 01<sub>B</sub>. This mode of operation is very similar to the previously discussed mode of operation [Chapter 25.2.1.2](#), with the difference that the Delta-sigma modulator is run at 160 MHz. Due to a lower data rate at the output of the Delta-sigma modulator, the CIC filter and the Compactor must be disabled. This mode is highlighted in [Figure 327](#) with a direct connection from the output of the modulator to the input of the multiplexer. During this mode of operation, the default update frequency of the loader  $f_{upd}$  is 1 MHz. User can also program the internal timer module [Chapter 25.2.2.1 CON](#).ITMDIV to change the update frequency. A  $f_{upd}$  of 10 MHz is not allowed in this mode.

*Note:* System SNR performance is limited in this mode.

### 25.2.2 HSPDM clocking and EVADC trigger generation

HSPDM receives two balanced and synchronized clocks of 320 MHz ( $f_{mod}$ ) and 160 MHz ( $f_{shift}$ ) from the CCU. For the shift register,  $f_{shift}$  is used to shift the data serially. For the Delta-sigma modulator signal path, the user can configure the modulator clock with 320 MHz or 160 MHz by selecting the appropriate mode using [CON.SM](#).

#### 25.2.2.1 Internal Timer Module (ITM)

The Internal Timer Module (ITM) generates the update frequency  $f_{upd}$  to control the loading of the 16-bit data from the SRAM. The ITM is a programmable clock divider which can be programmed to one of the possible four different divider values. These divider value can be configured using [CON.ITMDIV](#). Input clock to the ITM is always 160 MHz ( $f_{shift}$ ). The default value of the divider is 160, to support the default mode of operation of the bit-stream block [Chapter 25.2.1.2](#). [Table 1172](#) illustrates all the possible divider ratio which could be configured by the user. The user must read [Chapter 25.2.1](#) to set the correct divider value. The divider value, [CON.ITMDIV](#), must not be changed during the run time.

**Table 1172 Internal Timer Module Divider Value**

Bit Setting	Divider Value (ITMDIV)	$f_{upd}$
00 <sub>B</sub>	160	1 MHz
01 <sub>B</sub>	320	0.5 MHz
10 <sub>B</sub>	80	2 MHz
11 <sub>B</sub>	16	10 MHz

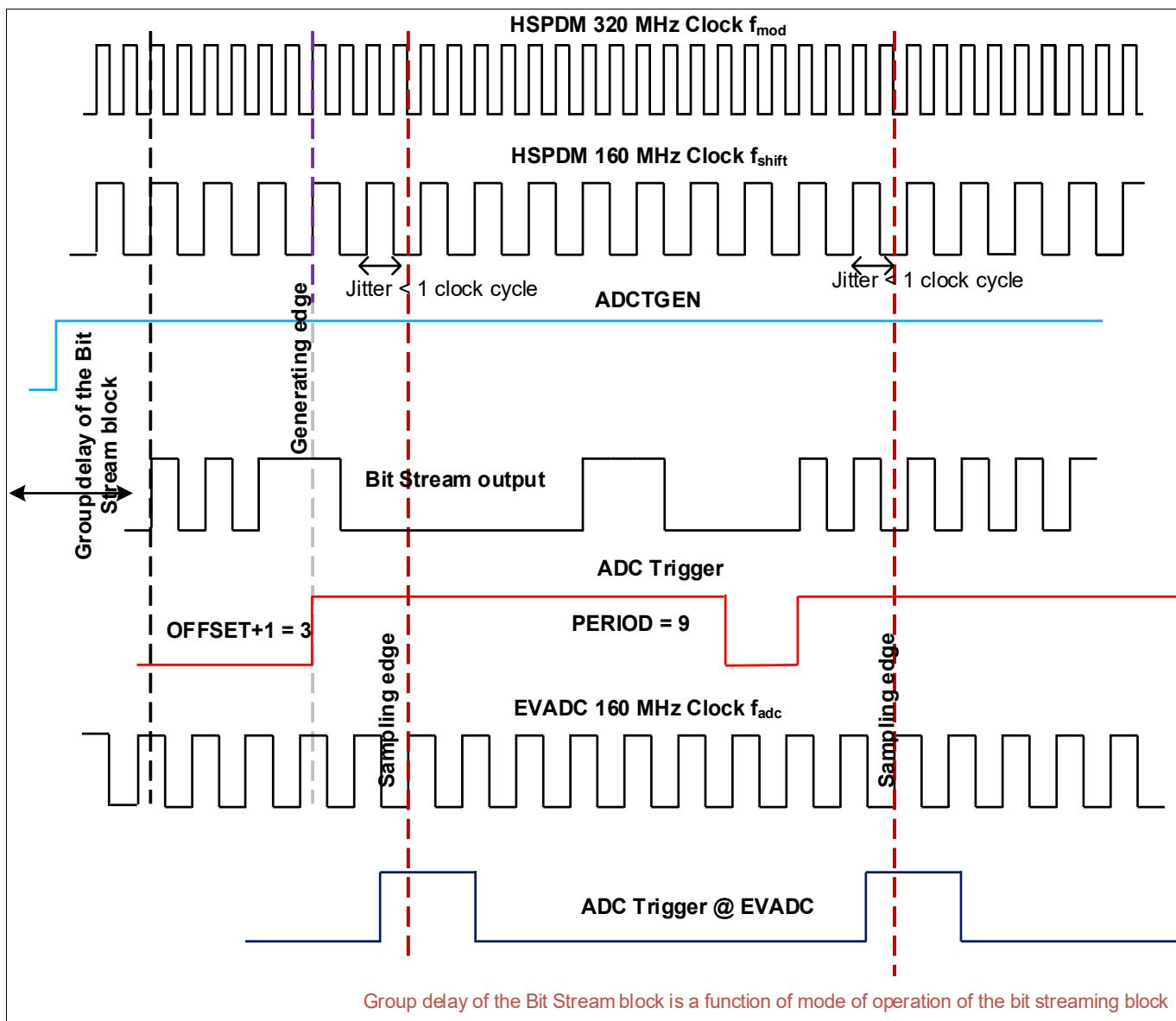
#### 25.2.2.2 ADC Trigger Generation

The ADC trigger generation block inside of the HSPDM ([Figure 326](#)) generates a signal to trigger to a group of EVADCs inside of the microcontroller, the start of conversion (SOC). The EVADC is used to convert the received analog signal on one of the analog input channels of the microcontroller into digital data. Using the [CON.ADCTGEN](#), the user can enable or disable the ADC trigger generation and configure the trigger signal generation using [ADCTG.OFFSET](#) and [ADCTG.PERIOD](#). The user can specify the offset value for the generation of the first trigger signal in [ADCTG.OFFSET](#) from the time of first output bit from the bit-stream block. This value can

## High Speed Pulse Density Modulation Module (HSPDM)

be between 0 and 409.59 us with a resolution step of 6.25 ns. The user can specify the period (after how much 160 MHz clock cycle must be the next trigger signal be generated) of the trigger signal using **ADCTG.PERIOD** field. This is a 16-bit down counter which after reaching zero, generates a trigger signal and starts counting again from the top. The total number of trigger signals can be configured using the **ADCTGCNT.TGCNT**. The total number of trigger signals is equal to **ADCTGCNT.TGCNT +1**. The maximum number of trigger signals which can be configured is 65536 ( $\text{FFFF}_H +1$ ). The pulse width of the trigger signal is fixed at eight cycles of 160 MHz clock. For **ADCTG.PERIOD<9**, there would only be one trigger signal with pulse width equal to  $\text{TGCNT} \times \text{PERIOD} +8$ . Setting of **ADCTG.PERIOD = 0** must be avoided.

**Figure 332** shows an example where **ADCTG.OFFSET =2**, that is, two clock cycles after the ADC trigger enable and the period of the ADC trigger signal is 9 clock cycles or 56.25 ns.

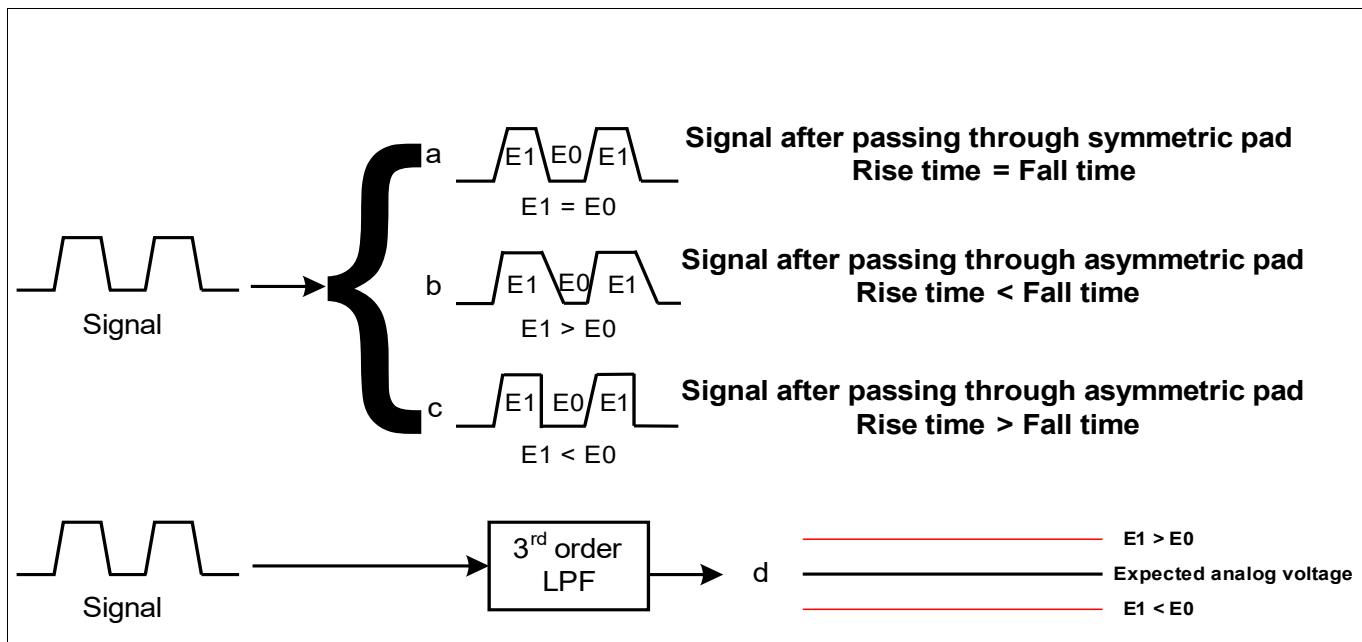


**Figure 332 ADC Trigger Concept**

### 25.2.3 Pad Asymmetry Compensation (PAC)

PAC is an optional feature included in HSPDM to give the user the possibility to easily predict the impact of pad asymmetries and compensate for it. Pad asymmetry degrades the SNR of the signal and also introduces an offset in the analog voltage.

## High Speed Pulse Density Modulation Module (HSPDM)



**Figure 333 (a) Symmetric pad (b) Asymmetric pad with the rise time < fall time (c) Asymmetric pad with the rise time> fall time (d) DC voltage offset after passing of the PDM through the LPF**

Every PDM bit-stream comprises of certain number of ones and zeros. When this bit-stream is passed through a low pass filter (LPF), an average voltage is created at the output of the LPF. This voltage is proportional to the number of ones and number of zeros. In reality the voltage is not only proportional to the number of ones and zeros, but also on the duty cycle of the signal. When the PDM signal is passed through a symmetric pad, the output signal has equal rise time and fall time and therefore, the duty cycle is maintained ([Figure 333\(a\)](#)). But pads are not symmetric and alter the rise time and fall time of the signal ([Figure 333\(a and b\)](#)). This signal when passed through a LPF ([Figure 333 \(d\)](#)) creates an offset from the expected analog voltage. The magnitude of this offset depends on the number of edges in the signal. The higher the number of edges, the higher is the impact of the pad asymmetries on the final voltage. Highest number of edges for a 2<sup>nd</sup> order Delta-sigma modulator can be seen in '110011001100...' output bit-stream. Therefore, the biggest offset is produced with such a bit-stream.

Referring to [Table 1169](#); the number of edges at the output of the Delta-sigma modulator increases from zero to maximum for input range 0000<sub>H</sub> to 8000<sub>H</sub> and decreases from maximum back to zero for input range 8000<sub>H</sub> to FFFF<sub>H</sub>. If there is a constant relation between the input to the Delta-sigma modulator and the number of edges in the PDM bit-stream at the output, then the user can easily compensate for offset voltage by applying a constant gain factor already to the 16-bit input word. PAC is intended to linearize the relationship between the input to the Delta-sigma modulator and the number of edges produced at the output.

PAC is implemented as a comparator inside the Delta-sigma modulator. The principle of the PAC is to increase the number of edges but not to alter the number of ones and zeros in the PDM bit-stream. By increasing the number of edges at the output, the offset which is produced after the LPF also increases. But this in turn gives a linear relation between the input word, the number of edges in the PDM bit-stream and also the final voltage offset after the LPF.

By default, the PAC is disabled inside the Delta-sigma modulator and the user shall enable it using **CON.PAC** bit ([Figure 328](#)). PAC is implemented as an algorithm which compares the output of the comparator, previous output of the Delta-sigma modulator and the PAC flip bit (pfb). The pfb ([Figure 328](#)) is used to flip the operation of the PAC; when the input to the Delta-sigma modulator is less than half of the full scale value (<2<sup>15</sup>) then the pfb is set to '1', and when the input is greater than or equal to half of the full scale value then the pfb is reset to '0'.

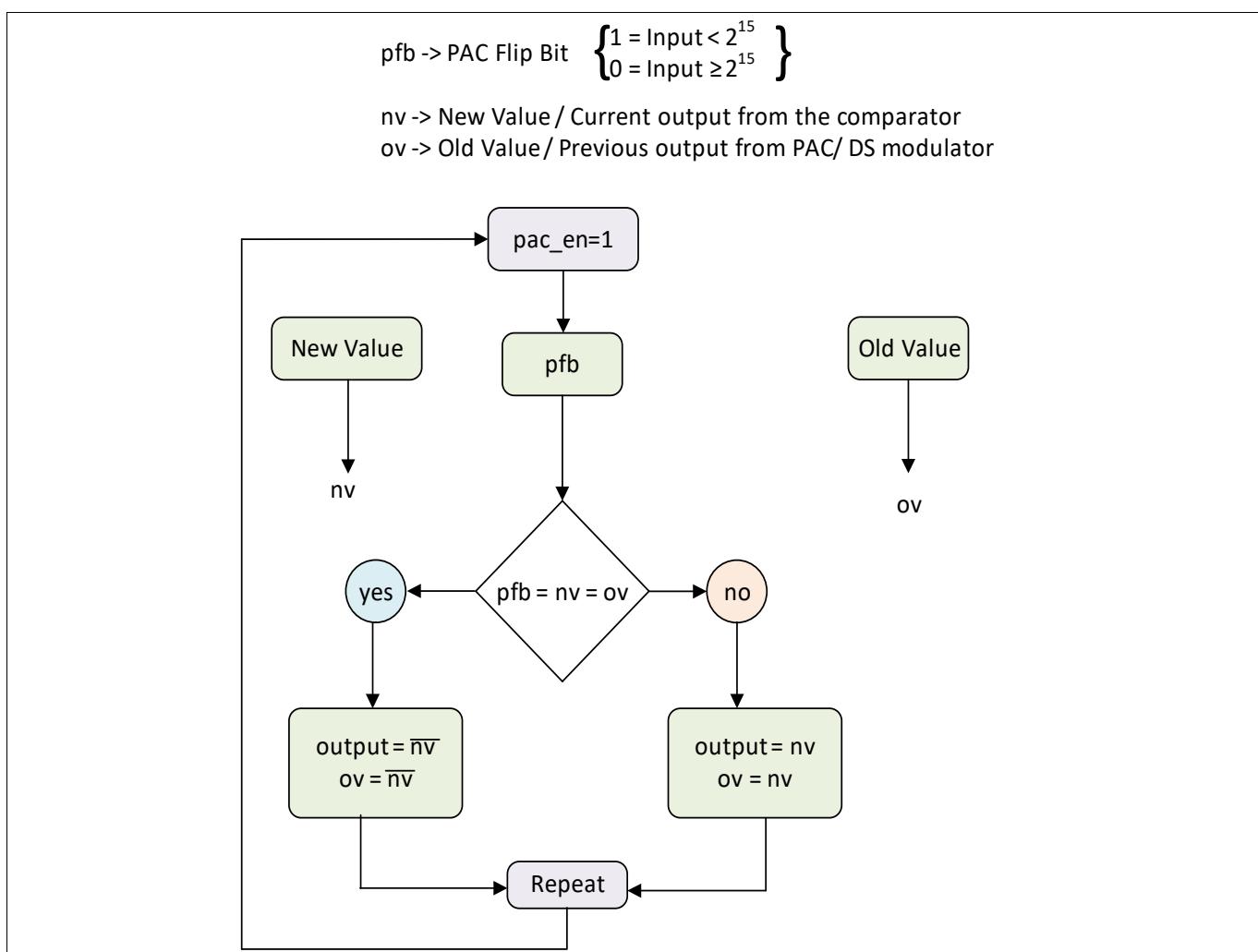
When the input is <2<sup>15</sup>, the PDM signal will have greater number of zeros than the number of ones and vice-versa for the input ≥ 2<sup>15</sup>. To increase the number of edges for the whole input range, PAC avoids the occurrence of two

## High Speed Pulse Density Modulation Module (HSPDM)

consecutive ones at the output of the Delta-sigma modulator for the input  $< 2^{15}$ , and occurrence of two consecutive zeros for the input  $\geq 2^{15}$ . Since the output value is fed back into the Delta-sigma loop, the loop corrects itself and maintains the pulse density of the signal. The PAC algorithm is explained in [Figure 334](#).

The PAC algorithm stores the previous output value (ov), which has the reset value '0'. When the PAC is enabled, the current output of the comparator (nv) is compared to the PAC flip bit (pfb) and ov. If these three values are the same then the negated nv value is pushed to the output and also stored as previous output value (ov). If the three compared values are not the same, then the output from the comparator is pushed to the output and the same value is stored as the old value, ov.

**Note:** *The user must not enable or disable the PAC during the run time. After the PAC is enabled or disabled, the Delta-sigma modulator must be restarted.*



**Figure 334** PAC algorithm

### 25.2.4 SRAM and Data Management

HSPDM has a 8KB in-built SRAM, which can store up to 4096 16-bit data points. The HSPDM RAM has an address space starting from  $F0280000_H$  to  $F0281FFF_H$  and the register address space starting from  $F0282000_H$  to  $F02820FF_H$  and support only word (32-bit) access. The SRAM and the RAM buffer manager (RAMBM) runs on the 100 MHz SPB (Serial Peripheral Bus) clock. The HSPDM supports two interface to the SPB, a Bus Peripheral Interface (BPI) for register access and a second slave interface SIF\_FPI to grant access to the SRAM [Figure 326](#).

## High Speed Pulse Density Modulation Module (HSPDM)

The SIF\_FPI interface supports burst mode for a fast upload of the SRAM. Although there are two interfaces to the SPB, the two interface can not be used in parallel.

Since the BSB0 and BSB1 are running on the other clock domain than the SRAM, there is a clock domain crossing from SRAM to bit-streaming blocks. Therefore, there is a synchronizing FIFO (sync FIFO) in between the SRAM and the bit-streaming blocks, which takes care of moving the 16-bit data from 100 MHz clock domain to bit-streaming block clock domain.

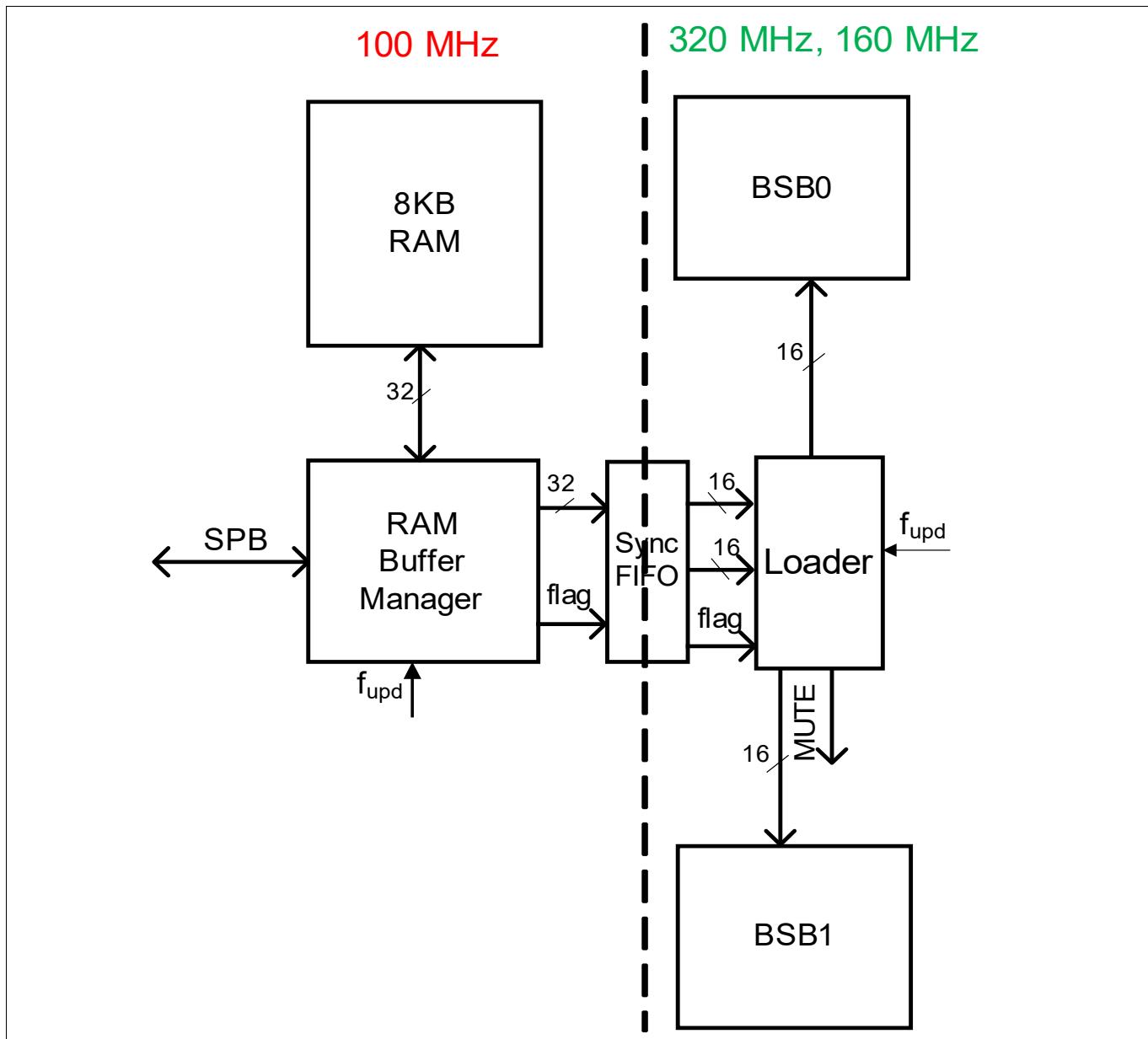


Figure 335 SRAM and data management

### 25.2.4.1 RAM Buffer Manager (RAMBM)

The RAM Buffer Manager, RAMBM, manages the following data tasks:

- Address generation for the RAM buffers that store the waveforms
- It defines two circular Buffers 'A' and 'B' with start and end addresses in the address space of the SRAM
- Manages switching between the Buffers 'A' and 'B'
- Transfers the data from the SRAM to the sync FIFO and manages to keep the sync FIFO constantly full

## High Speed Pulse Density Modulation Module (HSPDM)

- Controls the access to SRAM during the HSPDM run time
- Raise the flag on valid address matching between the address ranges stored in **MUTE0** and **MUTE1** and the current address which is fetched from the SRAM

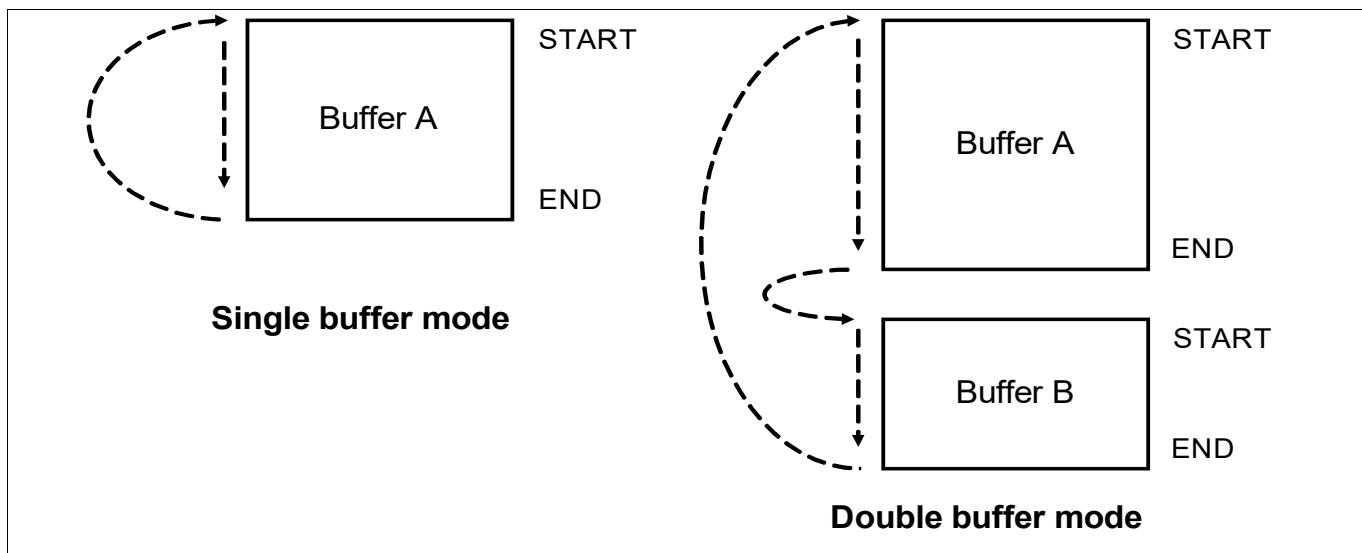
The registers **BUFA0** and **BUFB0** specifies the start and end address of Buffer A and Buffer B, respectively. **CURRAD** specify the current address being processed inside SRAM.

There are two modes of operation, single buffer mode (SBM) and double buffer mode (DBM). With **CON.MM** user can select between SBM and DBM. In SBM, only Buffer A is used, with automatic wrap around. In DBM, the streaming jumps between Buffer A and Buffer B in round robin. **CON.RUNS** (re)starts the bit-streaming from Buffer A start address. After reaching Buffer A end address, the address jumps to Buffer A start address or Buffer B start address, depending on the mode of operation: SBM or DBM, see [Figure 336](#) and [Figure 337](#).

The DBM also allows to switch between several waveforms stored in the RAM only by changing the start and end addresses of the two buffers.

If the start address and the end address of a buffer are equal and the mode is SBM, a constant value is streamed.

**Note:** *The software has the responsibility to program the start and end addresses correctly, such that, the end address is not lower than the start address. Also it is prohibited to change the mode during the run time. The mode, SBM or DBM must be configured at the start.*



**Figure 336 Single buffer mode and Double buffer mode**

### 25.2.4.2 MUTE Signal Generation

There are two registers, **MUTE0** and **MUTE1** that contain addresses to set and clear the signal MUTE. Both registers are completely interchangeable, meaning that the addresses defined in them are not associated in hardware to the start and the end of the two memory buffers in any way. The user has to take care that all the address settings fit such that the address range specified in **MUTE0** and **MUTE1** are non overlapping.

Each time the address counter in the RAMBM matches one of the two register values in **MUTE0** and **MUTE1**, the MUTE signal gets set or reset depending on the polarity bit field. MUTE signal can not be disabled but the polarity of the MUTE signal can be set using **CON.MPOL**. The content of the address defined in the registers **MUTE0** and **MUTE1** must match content of the address defines in the single buffer mode or double buffer mode. The status of the MUTE signal can be monitored by SW in the MUTE flag in the register **FLAGS**. The main purpose of this signal is to mark events with a defined time relation to the generated signal sequence see [Figure 338](#).

## High Speed Pulse Density Modulation Module (HSPDM)

Note: The MUTE signal is set and cleared when the content of the address defined in the registers **MUTE0** and **MUTE1** is loaded into the shift register or the Delta-sigma modulator. When the HSPDM is stopped and restarted again, the MUTE signal keeps its level, to reset the level of the MUTE signal, the HSPDM must be reset and then started.

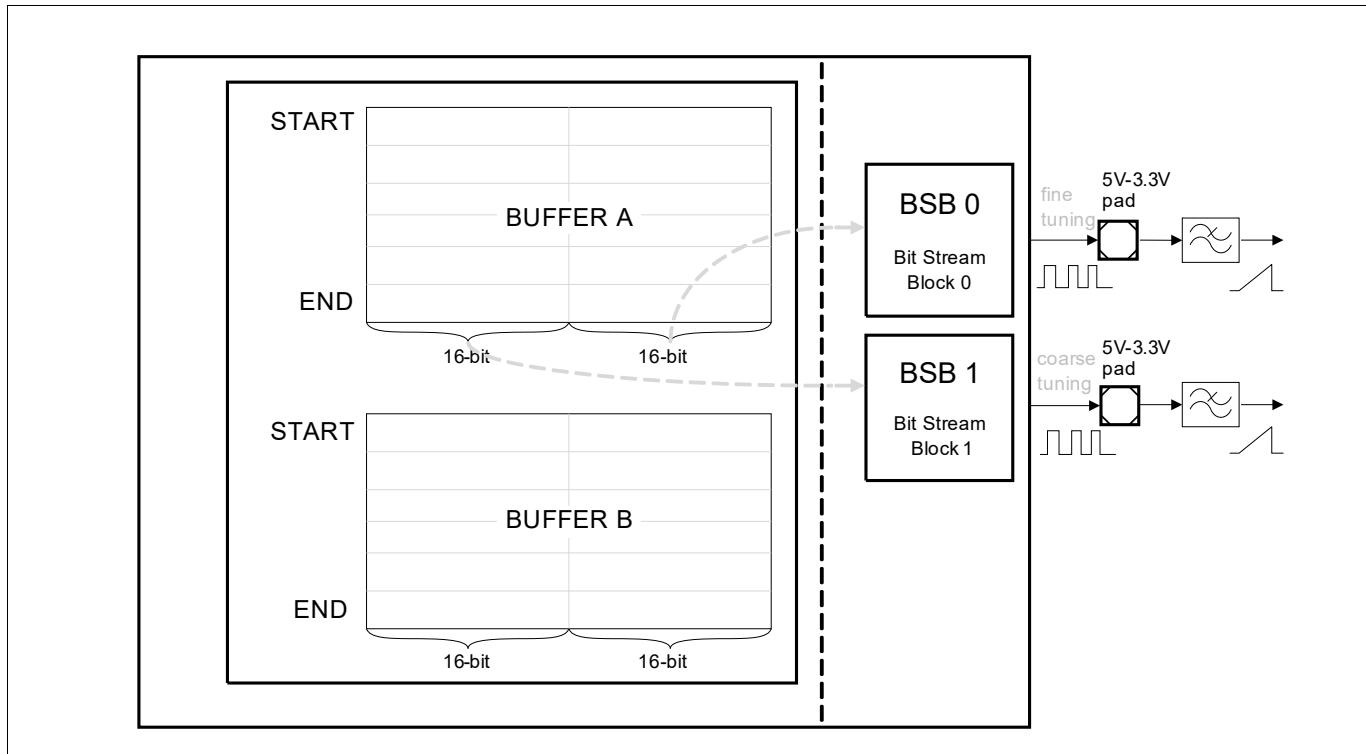
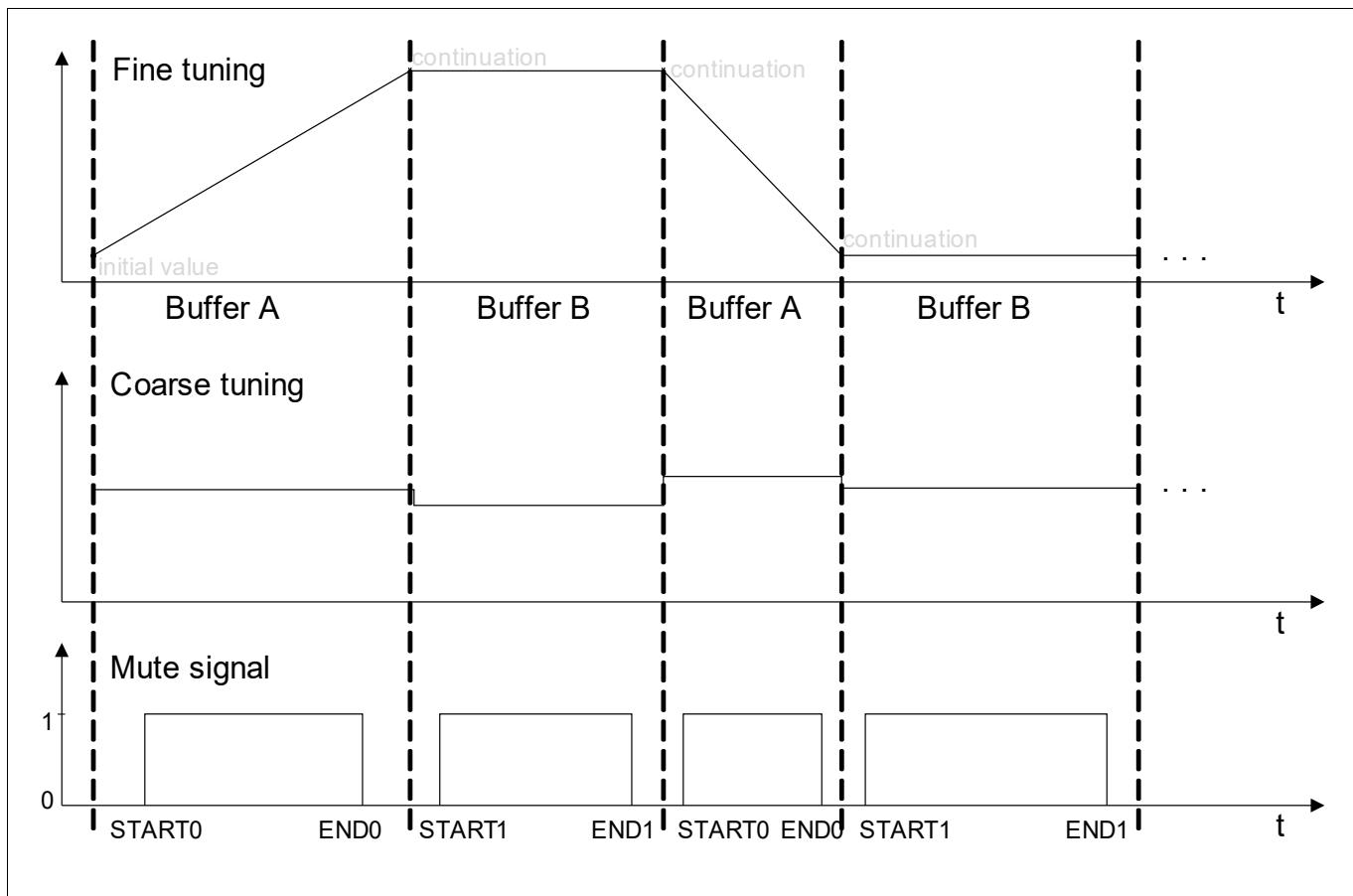


Figure 337 RAM storage of waveforms

## High Speed Pulse Density Modulation Module (HSPDM)



**Figure 338 Segmented Waveforms**

### 25.2.4.3 Interrupts

The HSPDM module generates the following interrupts:

- BFR, buffer interrupt, intended for triggering DMA transfers for filling the RAM
- MUTE, intended for notifying the CPU on certain events.
- Error interrupt, indicating RAM overflow.

BFR interrupt can be triggered by the buffer start and end events: BAS, BAE, BBS and BBE.

MUTE interrupt can be triggered by the MUTE Start and End events: M0S, M0E, M1S and M1E.

RAM overflow occurs if the last address of the RAM has been reached and a read from the next higher address has been attempted. It can occur if the start and the end addresses are wrongly configured (end less than start).

The register **FLAGS** contains the corresponding flags for these events. All events enabled in the register **FLAGSEN** trigger an interrupt. The flags can be cleared and set by software using the registers **FLAGSCLEAR** and **FLAGSSET**. Writing a '1' to a bit in the **FLAGSSET** register by software triggers the corresponding interrupt, if enabled.

### 25.2.4.4 Starting and Stopping the Bit-Streaming

HSPDM can be started or stopped by software or by hardware. The software can start the module by setting the RUN bit using **CON.RUNS** and the hardware by using the HWRUN signal. The status of the HSPDM can be monitored by reading the **CON.RUN** bit.

Each bit-stream block can be separately enabled or disabled by using the bits **CON.EN0** and **CON.EN1**. The user must not enable or disable the bit-stream block during the run. After changing the **CON.EN0** or **CON.EN1** bit, the HSPDM must be restarted. The user must avoid disabling both the bit-stream blocks. In such a use case user must

---

## High Speed Pulse Density Modulation Module (HSPDM)

stop the HSPDM. In the case of only one bit-streaming block enable, the SRAM operation remain the same as with the both bit-streaming blocks enabled. The upper or the lower 16-bits corresponding to the disabled bit-streaming block are ignored.

When a bit-streaming block is disabled or stopped, it drives 0 to the pad and all the internal states of the Delta-sigma modulator, the Shift register, the CIC filter and the compactor are reset to '0'.

### 25.2.4.5 Hardware Run Feature

The Hardware Run Feature allows starting the HSPDM with an edge of an external signal HWRUN. The software can select the type of the Active Edge (rising or falling) by using the bit **CON.HRAE**. Once started, the module runs independently of the signal HWRUN until the software clears the **CON.RUN** bit by using the bit **CON.RUNC**. After this the HWRUN signal can start the module again with new active edge.

If the bits RUNS and RUNC are erroneously both written with 1 at the same time by software, the RUN bit is cleared, meaning that RUNC has higher priority than RUNS. If the software wants to stop the module and the hardware to start the module, stopping the module via RUNC has higher priority.

---

## High Speed Pulse Density Modulation Module (HSPDM)

### 25.3 Registers

This section describes the kernel registers of the HSPDM module. All HSPDM kernel register names described in this section will be referenced in other parts by the module name prefix “HSPDM\_”.

All registers in the HSPDM address spaces are reset with the application reset.

The following tables give the overview of the HSPDM base addresses and registers.

HSPDM RAM takes 8KB, address space starting from  $F0280000_H$  to  $F0281FFF_H$ .

HSPDM register takes 256B, address space starting from  $F0282000_H$  to  $F02820FF_H$ .

## High Speed Pulse Density Modulation Module (HSPDM)

**Table 1173 Register Overview - HSPDM (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
CLC	Clock Control Register	0000 <sub>H</sub>	U,SV	SV,E,P	Application Reset	<b>29</b>
ID	Module Identification Register	0008 <sub>H</sub>	U,SV	BE	Application Reset	<b>18</b>
BUFA0	RAM Buffer A Register 0	0010 <sub>H</sub>	U,SV	U,SV,P	Application Reset	<b>18</b>
BUFB0	RAM Buffer B Register 0	0018 <sub>H</sub>	U,SV	U,SV,P	Application Reset	<b>19</b>
CURRAD	Current Address Register	001C <sub>H</sub>	U,SV	U,SV,P	Application Reset	<b>19</b>
MUTE0	MUTE0 Register	0020 <sub>H</sub>	U,SV	U,SV,P	Application Reset	<b>20</b>
MUTE1	MUTE1 Register	0024 <sub>H</sub>	U,SV	U,SV,P	Application Reset	<b>20</b>
ADCTG	ADC Trigger Register	0030 <sub>H</sub>	U,SV	U,SV,P	Application Reset	<b>21</b>
ADCTGCNT	ADC Trigger Count Register	0034 <sub>H</sub>	U,SV	U,SV,P	Application Reset	<b>22</b>
CON	Configuration Register	0038 <sub>H</sub>	U,SV	U,SV,P	Application Reset	<b>22</b>
FLAGS	Flags Register	0044 <sub>H</sub>	U,SV	SV,P	Application Reset	<b>24</b>
FLAGSSET	Flags Set Register	0048 <sub>H</sub>	U,SV	U,SV,P	Application Reset	<b>25</b>
FLAGSCLEAR	Flags Clear Register	004C <sub>H</sub>	U,SV	U,SV,P	Application Reset	<b>26</b>
FLAGSEN	Flags Enable Register	0050 <sub>H</sub>	U,SV	U,SV,P	Application Reset	<b>27</b>
OCS	OCDS Control and Status Register	00E8 <sub>H</sub>	U,SV	SV,P,OEN	See page <b>29</b>	<b>29</b>
KRSTCLR	Kernel Reset Status Clear Register	00EC <sub>H</sub>	U,SV	SV,P	Application Reset	<b>33</b>
KRST1	Kernel Reset Register 1	00F0 <sub>H</sub>	U,SV	SV,P	Application Reset	<b>33</b>
KRST0	Kernel Reset Register 0	00F4 <sub>H</sub>	U,SV	SV,P	Application Reset	<b>32</b>
ACCEN1	Access Enable Register 1	00F8 <sub>H</sub>	U,SV	BE	Application Reset	<b>31</b>
ACCENO	Access Enable Register 0	00FC <sub>H</sub>	U,SV	SV,SE	Application Reset	<b>31</b>

---

**High Speed Pulse Density Modulation Module (HSPDM)****List of Access Protection Abbreviations**

U - User Mode  
SV - Supervisor Mode  
BE - Bus Error  
nBE - no Bus Error  
P - Access Protection, as defined by the ACCEN Register  
E - ENDINIT  
SE - SafetyENDINIT

## High Speed Pulse Density Modulation Module (HSPDM)

### 25.3.1 Kernel Registers

#### Module Identification Register

The Module Identification Register ID contains read-only information about the module version.

ID															
Module Identification Register (0008 <sub>H</sub> ) Application Reset Value: 00EB C0XX <sub>H</sub>															
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16															
MODNUMBER															
r															
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0															
MODTYPE								MODREV							
r															

Field	Bits	Type	Description
MODREV	7:0	r	<b>Module Revision Number</b> MODREV defines the module revision number. The value of a module revision starts with 01 <sub>H</sub> (first revision).
MODTYPE	15:8	r	<b>Module Type</b> This bit field is C0 <sub>H</sub> . It defines a 32-bit module.
MODNUMBER	31:16	r	<b>Module Number Value</b> This bit field together with MODTYPE uniquely identifies a module. The MODNUMBER for this module is 00EB <sub>H</sub>

#### RAM Buffer A Register 0

The RAM Buffer A Register 0 contains information about the start and the end address of buffer 'A'. This register is read and write enable.

BUFA0																							
RAM Buffer A Register 0 (0010 <sub>H</sub> ) Application Reset Value: 0000 0000 <sub>H</sub>																							
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16																							
0								ENDA															
r																							
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0																							
0								STARTA															
r																							

Field	Bits	Type	Description
STARTA	12:0	rw	<b>Start Address of Buffer A</b> Word aligned address (the two LSB bits are always zero).

## High Speed Pulse Density Modulation Module (HSPDM)

Field	Bits	Type	Description
<b>ENDA</b>	28:16	rw	<b>End Address of Buffer A</b> Word aligned address (the two LSB bits are always zero).
<b>0</b>	15:13, 31:29	r	<b>Reserved</b> Read as 0; should be written with 0.

### RAM Buffer B Register 0

The RAM Buffer B Register 0 contains information about the start and the end address of buffer 'B'. This register is read and write enable.

#### BUFB0

<b>RAM Buffer B Register 0</b>																(0018 <sub>H</sub> )	<b>Application Reset Value: 0000 0000<sub>H</sub></b>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16																	

0	ENDB
r	rw

0	STARTB
r	rw

Field	Bits	Type	Description
<b>STARTB</b>	12:0	rw	<b>Start Address of Buffer B</b> Word aligned address (the two LSB bits are always zero).
<b>ENDB</b>	28:16	rw	<b>End Address of Buffer B</b> Word aligned address (the two LSB bits are always zero).
<b>0</b>	15:13, 31:29	r	<b>Reserved</b> Read as 0; should be written with 0.

### Current Address Register

The current address register contains information about the current address in the RAM. This register is read only.

#### CURRAD

<b>Current Address Register</b>																(001C <sub>H</sub> )	<b>Application Reset Value: 0000 0000<sub>H</sub></b>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16																	

0	CURRAD
r	rh

## High Speed Pulse Density Modulation Module (HSPDM)

Field	Bits	Type	Description
<b>CURRAD</b>	12:0	rh	<b>Current Address in RAM</b> Word aligned address (the two LSB bits are always zero)
<b>0</b>	31:13	r	<b>Reserved</b> Read as 0; should be written with 0.

### MUTE0 Register

This register defines the first start address and end address for the MUTE signal. This register is read and write enable

#### MUTE0

##### MUTE0 Register

(0020<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>START0</b>	12:0	rw	<b>Start Address 0</b> Word aligned address (the two LSB bits are always zero). When the data corresponding to this address is transferred to the input of the Delta-sigma modulator or to the shift register, the Mute signal is set. If the START and the END bitfields define the same address, set wins.
<b>END0</b>	28:16	rw	<b>End Address 1</b> Word aligned address (the two LSB bits are always zero). When the data corresponding to this address is transferred to the input of the Delta-sigma modulator or to the shift register, the Mute signal is cleared. If START and END bitfields define the same address, set wins.
<b>0</b>	15:13, 31:29	r	<b>Reserved</b> Read as 0; should be written with 0.

### MUTE1 Register

This register defines the second set of start and end address for the MUTE signal. This register is read and write enable.

## High Speed Pulse Density Modulation Module (HSPDM)

### MUTE1

#### MUTE1 Register

(0024<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			0												END1
r															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		0													START1
r															rw

Field	Bits	Type	Description
START1	12:0	rw	<b>Start Address 1</b> Word aligned address (the two LSB bits are always zero). When the data corresponding to this address is transferred to the input of the Delta-sigma modulator or to the shift register, the Mute signal is set. If the START and the END bitfields define the same address, set wins.
END1	28:16	rw	<b>End Address1</b> Word aligned address (the two LSB bits are always zero). When the data corresponding to this address is transferred to the input of the Delta-sigma modulator or to the shift register, the Mute signal is cleared. If the START and the END bitfields define the same address, set wins.
0	15:13, 31:29	r	<b>Reserved</b> Read as 0; should be written with 0.

### ADC Trigger Register

This register defines ADC trigger offset and the ADC trigger period.

### ADCTG

#### ADC Trigger Register

(0030<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
															PERIOD
															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															OFFSET
															rw

Field	Bits	Type	Description
OFFSET	15:0	rw	<b>Offset Delay from the Start of the Ramp</b> This bit field defines the time interval between the enabling of the trigger signal and the first ADC trigger event in the range of 0 to 65535 times the $f_{shift}$ period. The maximum configurable offset is 409.59 us.

## High Speed Pulse Density Modulation Module (HSPDM)

Field	Bits	Type	Description
<b>PERIOD</b>	31:16	rw	<b>PERIOD of the ADC Trigger Signal</b> This bit field defines the period of the ADC trigger signal. User can specify the period between 0 to 65535 times the $f_{shift}$ period. PERIOD=0 must be avoided

### ADC Trigger Count Register

This register defines the number of ADC trigger event in a single run

#### ADCTGCNT

#### ADC Trigger Count Register

(0034<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TGCNT															
rw															

Field	Bits	Type	Description
<b>TGCNT</b>	15:0	rw	<b>Number of Trigger Signals in a Single Ramp</b> This bit field defines the number of trigger signals in a single run. The number of trigger signals is equal to TGCNT +1. The maximum number of trigger signals is 65536. For PERIOD <9, there would only be one trigger signal with a pulse width of TGCNT*PERIOD+8. PERIOD=0 must be avoided
<b>0</b>	31:16	r	<b>Reserved</b> Read as 0; should be written with 0.

### Configuration Register

This register defines different configurations for the HSPDM module. Except the dither level change, the configuration must not be changed during the run. Every configuration change must be followed up with the HSPDM restart.

#### CON

#### Configuration Register

(0038<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

## High Speed Pulse Density Modulation Module (HSPDM)

Field	Bits	Type	Description
<b>EN0</b>	0	rw	<b>Enable Bit Streaming Block BSB 0</b> $0_B$ disabled $1_B$ enabled
<b>EN1</b>	1	rw	<b>Enable Bit Streaming Block BSB 1</b> $0_B$ disabled $1_B$ enabled
<b>SM</b>	3:2	rw	<b>Streaming Mode: Direct Shifting or Sigma-Delta Mode</b> $00_B$ Delta-sigma generated bit-stream with the CIC filter and the Compactor enabled (default) $01_B$ Delta-sigma generated bit-stream with the CIC filter and the Compactor disabled $10_B$ Shift register generated bit-stream $11_B$ Reserved
<b>PAC</b>	4	rw	<b>PAC enable or disable</b> $0_B$ PAC disable (default) $1_B$ PAC enable
<b>ITMDIV</b>	6:5	rw	<b>ITM divider value</b> $00_B$ 160 (default) $01_B$ 320 $10_B$ 80 $11_B$ 16
<b>MM</b>	7	rw	<b>Memory Mode</b> Selects between SBM and DBM $0_B$ SBM $1_B$ DBM
<b>RUN</b>	8	rh	<b>Run bit</b> Shows if the bit-streaming is started or stopped. It can be set and cleared by software using the RUNS and RUNC bits. It can also be set (but not cleared) by an edge of the hardware signal HWRUN. The active edge of the HWRUN signal is selected by using the bit HRAE. In case of software stopping the HSPDM module by writing to RUNC, and at the same time HWRUN starting the module, the software has higher priority. $0_B$ HSPDM stopped $1_B$ HSPDM running
<b>MPOL</b>	9	rw	<b>Mute Polarity</b> Configures the property of the Mute signal: active high or active low. Default 0 is active high. $0_B$ active high $1_B$ active low
<b>ADCTGEN</b>	10	rw	<b>ADC Trigger Block enable or disable</b> $0_B$ ADC Trigger disable $1_B$ ADC Trigger enable

## High Speed Pulse Density Modulation Module (HSPDM)

Field	Bits	Type	Description
DITH	13:11	rw	<b>Dither levels</b> $000_B$ disabled (default) $001_B$ Minimum dither level $010_B$ Low dither level $011_B$ Low-medium dither level $100_B$ Medium dither level $101_B$ Medium-high dither level $110_B$ High dither level $111_B$ Highest dither level
RUNC	16	w	<b>Run Bit Clear</b> Stops the bit-streaming. $0_B$ no action $1_B$ clear
RUNS	17	w	<b>Run Bit Set</b> Starts the bit-streaming (continue or re-start). $0_B$ no action $1_B$ set
HREN	18	rw	<b>Hardware Run Signal Enable</b> Enables the function of the HWRUN signal. $0_B$ disabled $1_B$ enabled
HRAE	19	rw	<b>Hardware Run Active Edge Selection</b> Selects if rising or falling edge starts the HSPDM module. $0_B$ rising edge $1_B$ falling edge
HRSEL	20	rw	<b>Hardware Run Input Selection</b> Selects the source for the HWRUN signal. $0_B$ CCU6 (default) $1_B$ Reserved
0	15:14, 31:21	r	<b>Reserved</b> Read as 0; should be written with 0.

### Flags Register

Contains flag bits for the corresponding events. Setting of a flag has a higher priority over the clearing of a flag.

#### FLAGS

##### Flags Register

(0044<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						MUTE	ER	M1E	M1S	MOE	MOS	BBE	BBS	BAE	BAS
r						rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

## High Speed Pulse Density Modulation Module (HSPDM)

Field	Bits	Type	Description
<b>BAS</b>	0	rh	<b>Buffer A Start Flag</b> Flags a read from the Buffer A start Address. Set by hardware, must be cleared by software.
<b>BAE</b>	1	rh	<b>Buffer A End Flag</b> Flags a read from the Buffer A end Address. Set by hardware, must be cleared by software.
<b>BBS</b>	2	rh	<b>Buffer B Start Flag</b> Flags a read from the Buffer B start Address. Set by hardware, must be cleared by software.
<b>BBE</b>	3	rh	<b>Buffer B End Flag</b> Flags a read from the Buffer B end Address. Set by hardware, must be cleared by software.
<b>M0S</b>	4	rh	<b>Mute 0 start flag</b> Flags a read from the MUTE0 start address. Set by hardware, must be cleared by software.
<b>M0E</b>	5	rh	<b>Mute 0 end flag</b> Flags a read from the MUTE0 end address. Set by hardware, must be cleared by software.
<b>M1S</b>	6	rh	<b>Mute 1 start flag</b> Flags a read from the MUTE1 start address. Set by hardware, must be cleared by software.
<b>M1E</b>	7	rh	<b>Mute 1 end flag</b> Flags a read from the MUTE1 end address. Set by hardware, must be cleared by software.
<b>ER</b>	8	rh	<b>Error RAM Overflow</b> Flags RAM overflow error. Set by hardware, must be cleared by software.
<b>MUTE</b>	9	rh	<b>Mute signal status</b> Shows the level of the MUTE signal. Set and reset by the hardware.
<b>0</b>	31:10	r	<b>Reserved</b> Read as 0; should be written with 0.

### Flags Set Register

Contains set bits for the corresponding flags. The bits are write only. Writing 1 performs the operation, writing 0 has no effect.

## High Speed Pulse Density Modulation Module (HSPDM)

## FLAGSSET

## Flags Set Register

(0048<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0																
r																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0								ER	M1E	M1S	MOE	MOS	BBE	BBS	BAE	BAS
r								W	W	W	W	W	W	W	W	

Field	Bits	Type	Description
<b>BAS</b>	0	w	<b>Buffer A Start Flag, Set Bit</b> Sets the corresponding flag.
<b>BAE</b>	1	w	<b>Buffer A End Flag, Set Bit</b> Sets the corresponding flag.
<b>BBS</b>	2	w	<b>Buffer B Start Flag, Set Bit</b> Sets the corresponding flag.
<b>BBE</b>	3	w	<b>Buffer B End Flag, Set Bit</b> Sets the corresponding flag.
<b>MOS</b>	4	w	<b>Mute 0 Start Flag, Set Bit</b> Sets the corresponding flag.
<b>MOE</b>	5	w	<b>Mute 0 End Flag, Set Bit</b> Sets the corresponding flag.
<b>M1S</b>	6	w	<b>Mute 1 Start Flag, Set Bit</b> Sets the corresponding flag.
<b>M1E</b>	7	w	<b>Mute 1 End Flag, Set Bit</b> Sets the corresponding flag.
<b>ER</b>	8	w	<b>Error RAM Overflow, Set Bit</b> Sets the corresponding flag.
<b>0</b>	31:9	r	<b>Reserved</b> Read as 0; should be written with 0.

## Flags Clear Register

Contains clear bits for the corresponding flags. The bits are write only. Writing 1 performs the operation, writing 0 has no effect.

## High Speed Pulse Density Modulation Module (HSPDM)

### FLAGSCLEAR

#### Flags Clear Register

(004C<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0																
r																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0								ER	M1E	M1S	MOE	MOS	BBE	BBS	BAE	BAS
w								w	w	w	w	w	w	w	w	

Field	Bits	Type	Description
<b>BAS</b>	0	w	<b>Buffer A Start Flag, Clear Bit</b> Clears the corresponding flag.
<b>BAE</b>	1	w	<b>Buffer A End Flag, Clear Bit</b> Clears the corresponding flag.
<b>BBS</b>	2	w	<b>Buffer B Start Flag, Clear Bit</b> Clears the corresponding flag.
<b>BBE</b>	3	w	<b>Buffer B End Flag, Clear Bit</b> Clears the corresponding flag.
<b>MOS</b>	4	w	<b>Mute 0 Start Flag, Clear Bit</b> Clears the corresponding flag.
<b>MOE</b>	5	w	<b>Mute 0 End Flag, Clear Bit</b> Clears the corresponding flag.
<b>M1S</b>	6	w	<b>Mute 1 Start Flag, Clear Bit</b> Clears the corresponding flag.
<b>M1E</b>	7	w	<b>Mute 1 End Flag, Clear Bit</b> Clears the corresponding flag.
<b>ER</b>	8	w	<b>Error RAM Overflow, Clear Bit</b> Clears the corresponding flag.
<b>0</b>	31:9	r	<b>Reserved</b> Read as 0; should be written with 0.

### Flags Enable Register

Contains enable bits for the interrupt on the corresponding events.

## High Speed Pulse Density Modulation Module (HSPDM)

### FLAGSEN

#### Flags Enable Register

(0050<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0																
r																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0								ER	M1E	M1S	MOE	MOS	BBE	BBS	BAE	BAS
r								rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
<b>BAS</b>	0	rw	<b>Buffer A Start, Enable Bit</b> Enables an interrupt on the corresponding event.
<b>BAE</b>	1	rw	<b>Buffer A End, Enable Bit</b> Enables an interrupt on the corresponding event.
<b>BBS</b>	2	rw	<b>Buffer B Start, Enable Bit</b> Enables an interrupt on the corresponding event.
<b>BBE</b>	3	rw	<b>Buffer B End, Enable Bit</b> Enables an interrupt on the corresponding event.
<b>MOS</b>	4	rw	<b>Mute 0 Start, Enable Bit</b> Enables an interrupt on the corresponding event.
<b>MOE</b>	5	rw	<b>Mute 0 End, Enable Bit</b> Enables an interrupt on the corresponding event.
<b>M1S</b>	6	rw	<b>Mute 1 Start, Enable Bit</b> Enables an interrupt on the corresponding event.
<b>M1E</b>	7	rw	<b>Mute 1 End, Enable Bit</b> Enables an interrupt on the corresponding event.
<b>ER</b>	8	rw	<b>Error RAM Overflow, Enable Bit</b> Enables an interrupt on the corresponding event.
<b>0</b>	31:9	r	<b>Reserved</b> Read as 0; should be written with 0.

### 25.3.2 BPI\_FPI Registers

#### BPI\_FPI Registers Overview

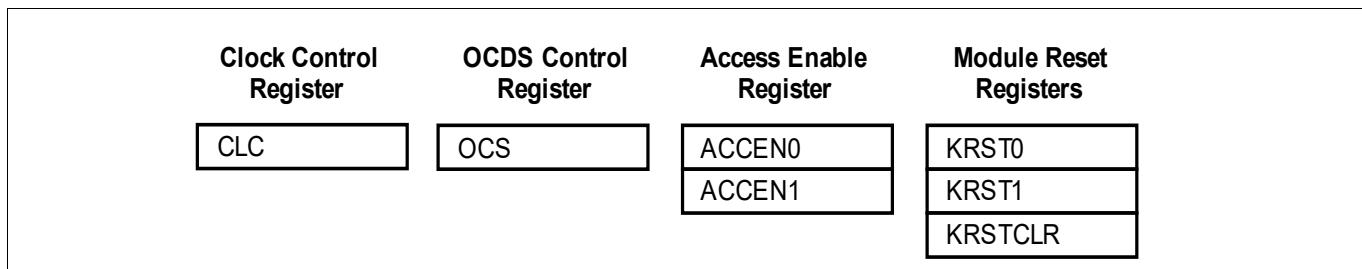


Figure 339 BPI\_FPI Registers

## High Speed Pulse Density Modulation Module (HSPDM)

**Figure 339** shows all registers associated with the BPI\_FPI module, configured for one kernel. The writes of the bus masters to the HSPDM module is controlled by Access Protection registers ACCENx.

The HSPDM implements two ACCENx registers, ACCEN0 and ACCEN1. The SRAM write access is also protected by Access Protection registers ACCENx.

The ACCENx registers are protected by Safety EndInit mechanism.

HSPDM must be stopped before entering the sleep mode and restarted after exiting the sleep mode. During the sleep mode, access to registers and SRAM is not allowed. It is not recommended to run HSPDM in sleep mode.

HSPDM does not support any soft suspend mode although there is a bit **OCS.SUS**. The **OCS.SUS** is not connected to any control of the HSPDM.

HSPDM supports hard suspend using **OCS.SUS**. In this mode, the clocks to the HSPDM are switched off, but registers are available for a read. During the hard suspend a read or a write performed on the HSPDM SRAM will result in a bus error. After the hard suspend, the user must make sure to reset the HSPDM. The behavior is undefined if there is no reset following the end of hard suspend.

### Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI\_FPI for the module. Where a module kernel is connected to the CLC clock control interface, CLC controls the module clock signal, sleep mode and disable mode for the module.

<b>CLC</b>															
<b>Clock Control Register</b>															
<b>(0000<sub>H</sub>) Application Reset Value: 0000 0003<sub>H</sub></b>															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
								r				rw	r	rh	rw

Field	Bits	Type	Description
<b>DISR</b>	0	rw	<b>Module Disable Request Bit</b> Used for enable/disable control of the module.
<b>DISS</b>	1	rh	<b>Module Disable Status Bit</b> Bit indicates the current status of the module.
<b>EDIS</b>	3	rw	<b>Sleep Mode Enable Control</b> Used to enable the module's sleep mode. $0_B$ Enabled $1_B$ Disabled
<b>0</b>	2, 31:4	r	<b>Reserved</b> Should be written with 0.

### OCDS Control and Status Register

The OCDS control and status register OCS controls the module's behavior in suspend mode (used for debugging).

## High Speed Pulse Density Modulation Module (HSPDM)

The OCDS control and status register is cleared by debug reset. The register can only be written when the OCDS is enabled. When OCDS is disabled the OCS suspend control is ineffective.

### OCS

#### OCDS Control and Status Register

( $00E8_H$ )

**Reset Value:** [Table 1175](#)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	<b>SUSST A</b>	<b>SUS_P</b>			<b>SUS</b>						0				
r	rh	w		rw							r				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							<b>0</b>								r

Field	Bits	Type	Description
<b>SUS</b>	27:24	rw	<b>OCDS Suspend Control</b> Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) $0_H$ Will not suspend $1_H$ Hard suspend. Clock is switched off immediately. $2_H$ Soft suspend <b>others</b> , reserved
<b>SUS_P</b>	28	w	<b>SUS Write Protection</b> SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
<b>SUSSTA</b>	29	rh	<b>Suspend State</b> Shows the current suspend state of the module. $0_B$ Module is not (yet) suspended $1_B$ Module is suspended
<b>0</b>	23:0, 31:30	r	<b>Reserved</b> Read as 0; must be written with 0. The bits [4:0] are rw (readable and writable).

**Table 1174 Access Mode Restrictions of OCS sorted by descending priority**

Mode Name	Access Mode		Description
otherwise	r	SUS	
write 1 to <b>SUS_P</b> (default)	rw	SUS	

**Table 1175 Reset Values of OCS**

Reset Type	Reset Value	Note
PowerOn Reset	$0000\ 0000_H$	
Debug Reset	$0000\ 0000_H$	

## High Speed Pulse Density Modulation Module (HSPDM)

### Access Enable Register 0

The Access Enable Register 0 controls write access<sup>1)</sup> for transactions with the on chip bus master TAG ID 000000<sub>B</sub> to 011111<sub>B</sub>. The BPI\_FPI is prepared for a 6-bit TAG ID. The registers ACCENO / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCENO.ENx: EN0 -> TAG ID 000000<sub>B</sub>, EN1 -> TAG ID 000001<sub>B</sub>, ..., EN31 -> TAG ID 011111<sub>B</sub>.

#### ACCENO

##### Access Enable Register 0

(00FC<sub>H</sub>)Application Reset Value: FFFF FFFF<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rw															

Field	Bits	Type	Description
ENy (y=0-31)	y	rw	<b>Access Enable for Master TAG ID y</b> This bit enables write access to the module kernel addresses for transactions with the Master TAG ID y 0 <sub>B</sub> Write access will not be executed 1 <sub>B</sub> Write access will be executed

### Access Enable Register 1

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000<sub>B</sub> to 111111<sub>B</sub>. The BPI\_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000<sub>B</sub>, EN1 -> TAG ID 100001<sub>B</sub>, ..., EN31 -> TAG ID 111111<sub>B</sub>.

#### ACCEN1

##### Access Enable Register 1

(00F8<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								0							
								r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								0							
								r							

1) The BPI\_FPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

## High Speed Pulse Density Modulation Module (HSPDM)

Field	Bits	Type	Description
0	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

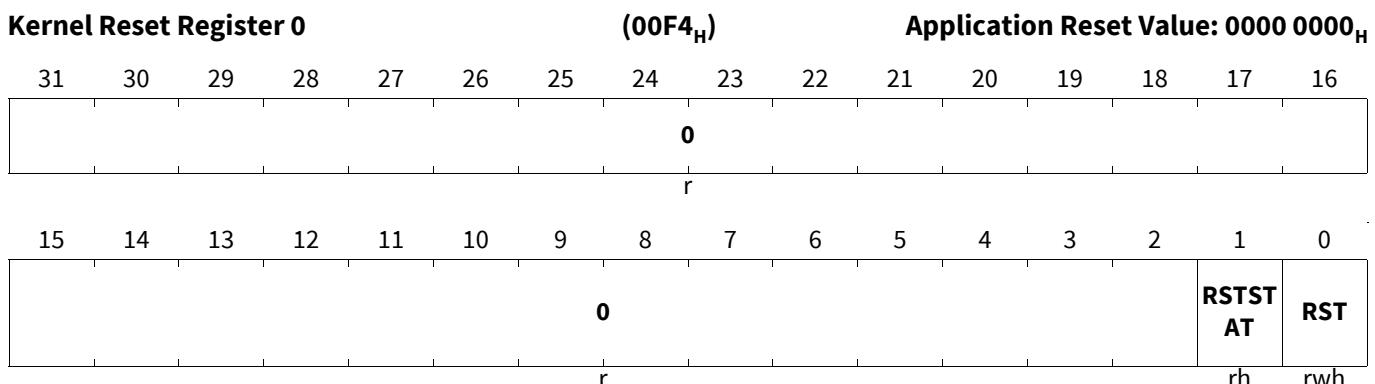
### Kernel Reset Register 0

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI\_FPI in the same clock cycle the RST bit is re-set by the BPI\_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLR.CLR register bit.

During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

#### KRST0



Field	Bits	Type	Description
RST	0	rwh	<b>Kernel Reset</b> This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed. 0 <sub>B</sub> No kernel reset was requested 1 <sub>B</sub> A kernel reset was requested
RSTSTAT	1	rh	<b>Kernel Reset Status</b> This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits. This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register. 0 <sub>B</sub> No kernel reset was executed 1 <sub>B</sub> Kernel reset was executed
0	31:2	r	<b>Reserved</b> Read as 0; should be written with 0.

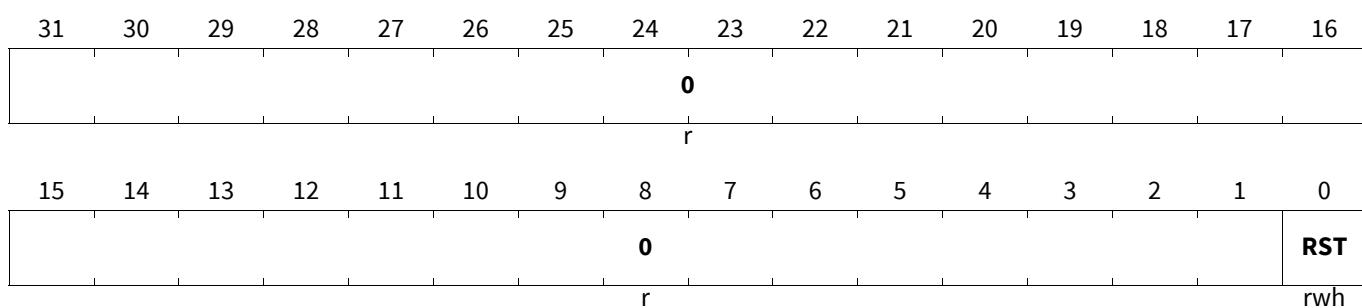
## High Speed Pulse Density Modulation Module (HSPDM)

### Kernel Reset Register 1

The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

#### KRST1

##### Kernel Reset Register 1

(00F0<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

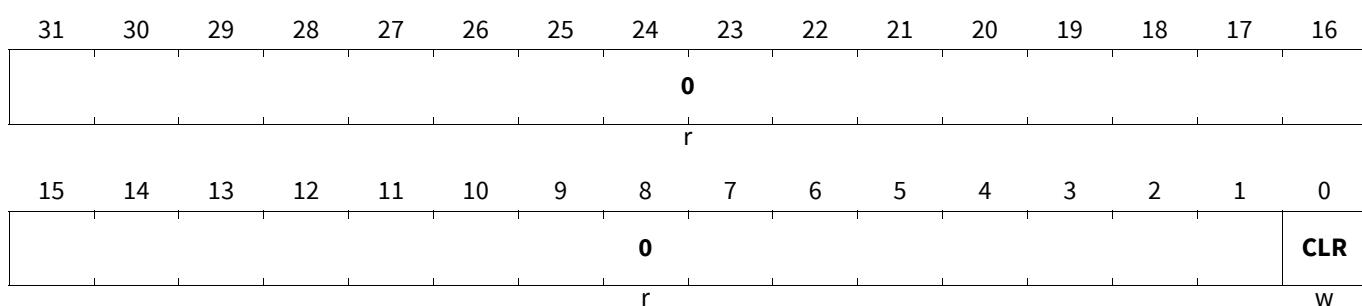
Field	Bits	Type	Description
RST	0	rwh	<b>Kernel Reset</b> This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed. 0 <sub>B</sub> No kernel reset was requested 1 <sub>B</sub> A kernel reset was requested
0	31:1	r	<b>Reserved</b> Read as 0; should be written with 0.

### Kernel Reset Status Clear Register

The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

#### KRSTCLR

##### Kernel Reset Status Clear Register

(00EC<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
CLR	0	w	<b>Kernel Reset Status Clear</b> Read always as 0. 0 <sub>B</sub> No action 1 <sub>B</sub> Clear Kernel Reset Status KRST0.RSTSTAT

## High Speed Pulse Density Modulation Module (HSPDM)

Field	Bits	Type	Description
0	31:1	r	<b>Reserved</b> Read as 0; should be written with 0.

## 25.4 IO Interfaces

**Table 1176 List of HSPDM Interface Signals**

Interface Signals	I/O	Constraints	Description
reset			
sx_clocks			
DTCM1			Scan control socket for SRI/SPB domain
DTCM3			Scan control socket for peripheral domain
protect			
sx_ocds_periph_ctrl			
sx_ssh_com			
hspdm			
ram			<b>FPI slave interface for SRAM access</b>
regs			<b>FPI slave interface for BPI registers access</b>
sx_irq_hspdm			
INT(2:0)	out		<b>HSPDM Service Request</b>
FPI_SLEEP	in		<b>SCU FPI sleep</b>
HWRUN(1:0)	in		<b>Hardware trigger inputs</b>
MUTE	out		<b>Mute output to tx</b>
BS0_OUT	out		<b>Bit stream 0 output</b>
BS1_OUT	out		<b>Bit stream 1 output</b>
BS0_PAD_IN	in		<b>Bit stream 0 feedback from pad</b>
BS1_PAD_IN	in		<b>Bit stream 1 feedback from pad</b>
ADC_TRIG_OUT	out		<b>Trigger output to ADC</b>

## 25.5 Revision History

**Table 1177 Revision History**

Reference	Change to Previous Version	Comment
V0.7.8		
<a href="#">Page 23</a>	Updated ADCTGEN in register CON	
V0.7.9		
<a href="#">Page 6</a>	Change the Figure 7 ADC trigger signal.	

**RESTRICTED**

**AURIX™ TC3xx**

**NDA Required**



---

**High Speed Pulse Density Modulation Module (HSPDM)**

---

## Camera and ADC Interface (CIF)

### 26 Camera and ADC Interface (CIF)

The Camera and ADC Interface Module (CIF) provides 16-bit wide parallel read interface that can be used to connect camera sensors and external Analog to Digital Converters (ADCs).

#### 26.1 Feature List

**The following list summarizes the CIF's features:**

- Throughput up to 96 MPixel/s
- 32-bit BBB slave programming interface
- BBB master interface
- ITU-R BT 601 compliant video interface supporting  $YC_bC_r$  and RAW data transfer
- ITU-R BT 656 compliant video interface supporting  $YC_bC_r$  and RAW data transfer
- Non line/frame aligned data transfer (data mode)
- 16-bit parallel camera and ADC interface
- $YC_bC_r$  4:2:2 processing
- Hardware JPEG encoder (supporting images up to a horizontal resolution of 1280 pixel) incl. JFIF1.02 stream generator and programmable quantization and Huffman tables
- Windowing and frame synchronization
- 1 Main and 5 Extra Image Cropping units allowing parallel transfer and position adjustment of up to 6 subregions
- Path from Main Image Stabilization or from one Extra Image Cropping unit to AGBT debug interface including a Metasymbol Generation unit inserting frame start and timing information into the stream
- Programmable watchdog timer to detect abortion/breaks in frame transmission
- Linear downscaling for the first extra path, supporting a mode for RGB Bayer Pattern
- Frame skip support for video (e.g. MPEG-4) encoding
- Macro block line, frame end, capture error, data loss interrupts and synchronization (h\_start, v\_start) interrupts
- Programmable polarity for synchronization signals
- Luminance/chrominance and chrominance blue/red swapping for  $YC_bC_r$  input signals
- Maximum input resolution of 4095x4095 pixels
- Buffer in EMEM organized as ring-buffer, supporting up to 2x8 Beat Bursts (2x32 Bytes)
- Buffer overflow protection for RAW data transfer and JPEG files
- Asynchronous reset input, software reset for the entire IP
- Support of planar, semi planar and interleaved storage format (main path)
- Power management by software controlled clock disabling of currently not needed sub-modules

#### 26.2 Overview

The following section provides overview of the architecture of the CIF module and its applications.

---

## Camera and ADC Interface (CIF)

### 26.2.1 Introduction

The Camera and ADC Interface (CIF) represents a complete video and still picture input interface transferring data from an image sensor into EMEM. Furthermore several hardware blocks - performing image processing operations on the incoming data - are provided.

#### 26.2.1.1 Camera and ADC Interface Functional Overview

Apart from providing the physical interfacing to various types of camera sensor modules, the CIF block implements image processing and encoding functionality. The integrated image processing unit supports image sensors with integrated  $Y\bar{C}_b\bar{C}_r$  processing. Additionally the CIF also supports transfer of RAW (e.g. Bayer Pattern) images and non frame synchronized data packets.

The CIF block features a 16 bit parallel interface.

All output data is transmitted via the memory interface to a BBB system using the master interface.

Programming of the CIF is done by register read/write transactions using a BBB slave interface.

#### 26.2.1.2 Camera and ADC Interface Block Diagram

[Figure 340](#) shows the \*\*\* 'Camera and ADC Interface Block Diagram' on page 2 \*\*\* (top level view).

The BBB interface is divided into a master and a slave interface with their own clock domains. These clock domains may be asynchronous to the CIF Module Clock. To avoid data loss the frequency of the BBB clocks need to fulfill the following requirements:

- BBB Interface Master Clock (DMA)  $\geq$  CIF Module Clock
- BBB Interface Master Clock (DMA)  $\leq 2 *$  CIF Module Clock
- CIF Module Clock  $\geq$  Pixel Clock (PCLK)

## Camera and ADC Interface (CIF)

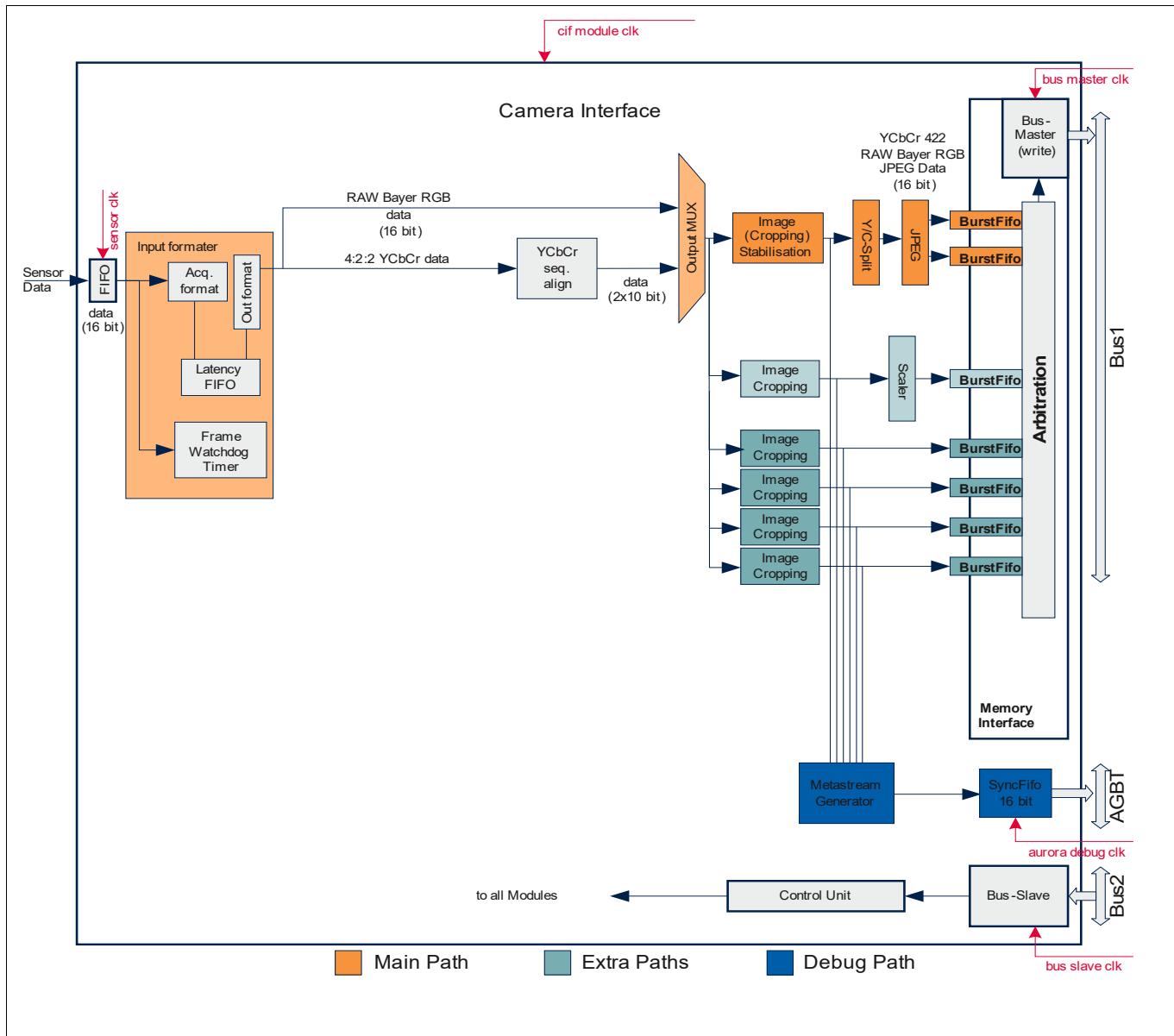


Figure 340 \*\*\* 'Camera and ADC Interface Block Diagram' on page 2 \*\*\*

### 26.2.2 Camera and ADC Interface Functional Specification

This chapter describes the CIF's functionality.

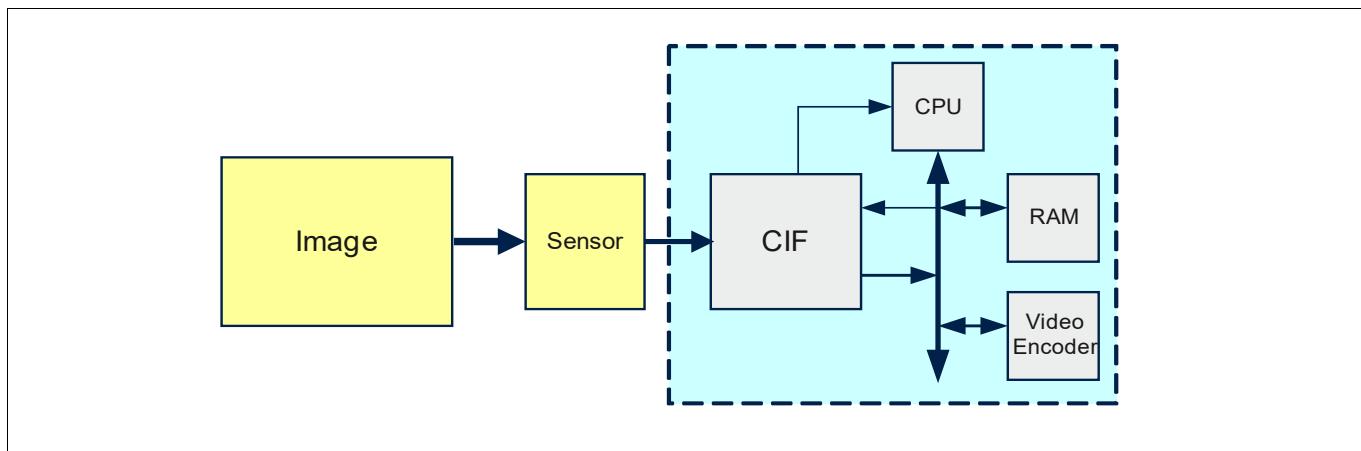
#### 26.2.2.1 Target Applications

The CIF is targeted on various applications requiring a mega pixel still image and/or video input (e.g.: mobile devices, automotive or industrial vision, ...).

##### 26.2.2.1.1 Camera Interface Example

A target application example is shown in [Figure 341](#).

## Camera and ADC Interface (CIF)



**Figure 341 Target application example**

The CIF provides a sensor/camera interface for a wide variety of video applications and it is optimized for high speed data transmission under terms of low power consumption.

**Table 1178 Available Signals for Sensor Interface**

Source	CIF I/O Name <sup>1)</sup>	Description
Sensor	PCLK	Pixel Clock from Sensor
Sensor	HSYNC	Horizontal Synchronisation Signal from Sensor
Sensor	VSYNC	Vertical Synchronisation Signal from Sensor
Sensor	D[15:0]	Data Bus from sensor (16 bit maximum)

1) See [Section 26.5, “IO Interfaces”](#)

This module is designed to be used for the following use cases:

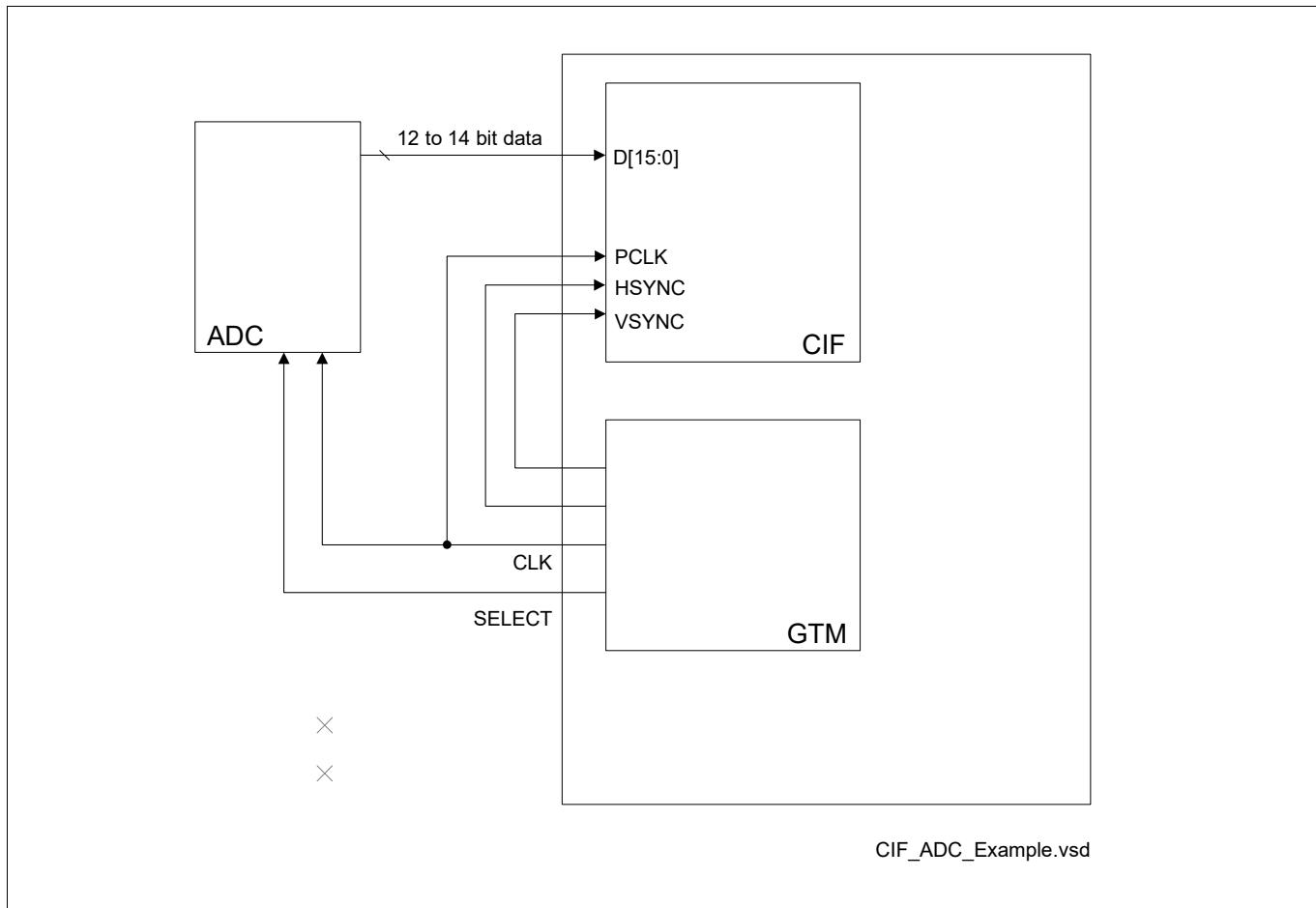
- Video capturing/encoding
- Still image capturing in  $YC_bC_r$  with on-the-fly JPEG encoding
- RAW frame data capturing
- Simple non line/frame aligned data reception (data mode) used for example for connecting to external ADCs

The CIF requires fast system memory for image storage in either planar/semi-planar/interleaved  $YC_bC_r$  or RAW planar format or as JPEG compressed data. The integrated JPEG encoding engine is able to generate a full JFIF 1.02 compliant JPEG file that can be displayed directly by any image viewer.

All important  $YC_bC_r$  formats - which are used for video compression (e.g. MPEG4) for instance - are supported. For on-the-fly encoding macro block line interrupts are generated to trigger video encoding.

### 26.2.2.1.2 Connecting External ADC

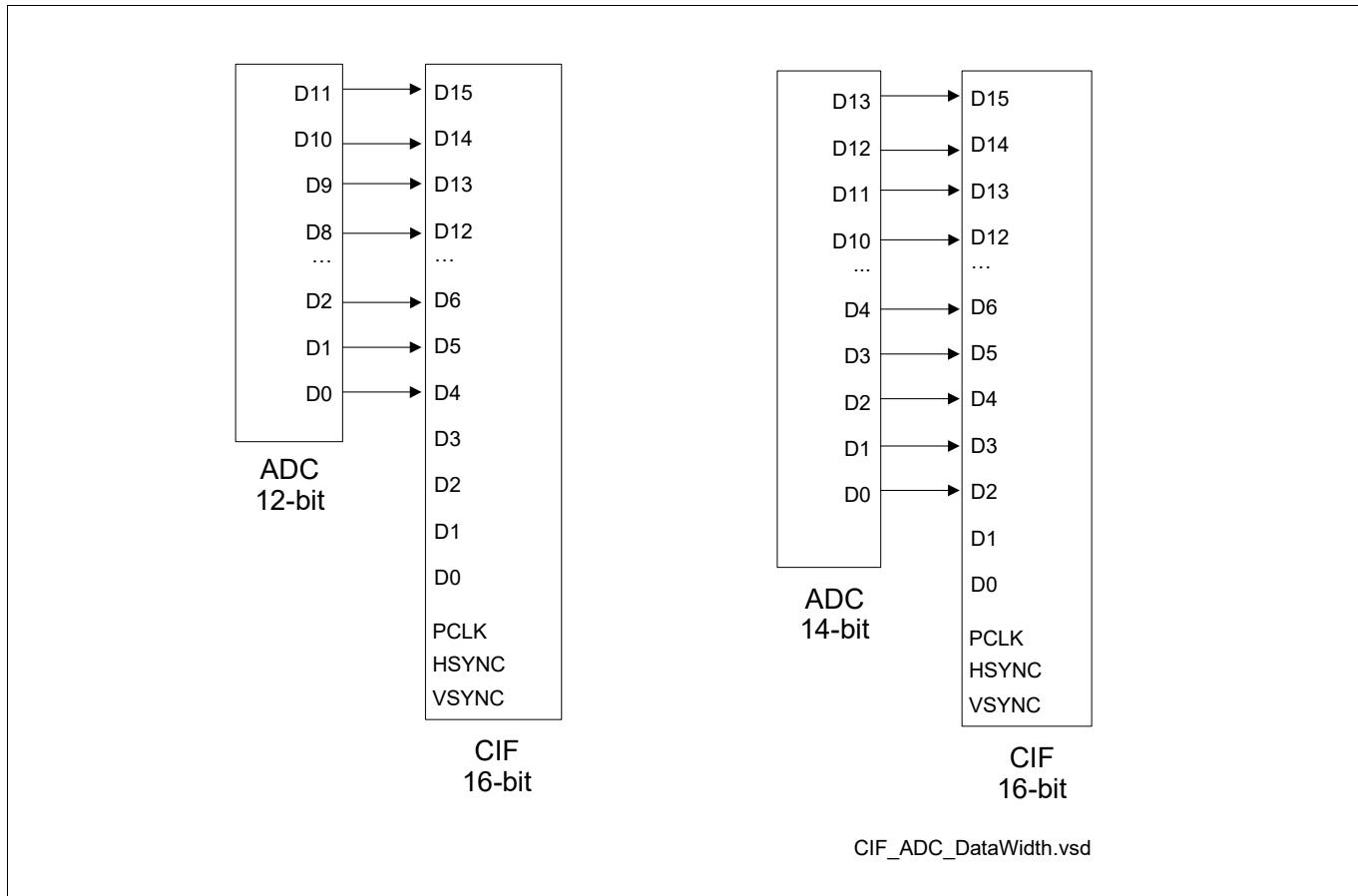
The CIF provides a parallel interface that can be used to connect external ADC converters. In order to connect to one or more external ADCs, the CIF module requires a timer module capable of generating the appropriate clock and select signals, like the Generic Timer Module (GTM). The [Figure 342](#) shows an example of a connection between a single ADC and the CIF module.

**Camera and ADC Interface (CIF)****Figure 342 Example of a Connection to an External ADC**

Connecting multiple ADCs to the CIF module may require an external multiplexer, and additional control signals to control it. Using twin or quad ADCs avoids the need for an external multiplexer.

## Camera and ADC Interface (CIF)

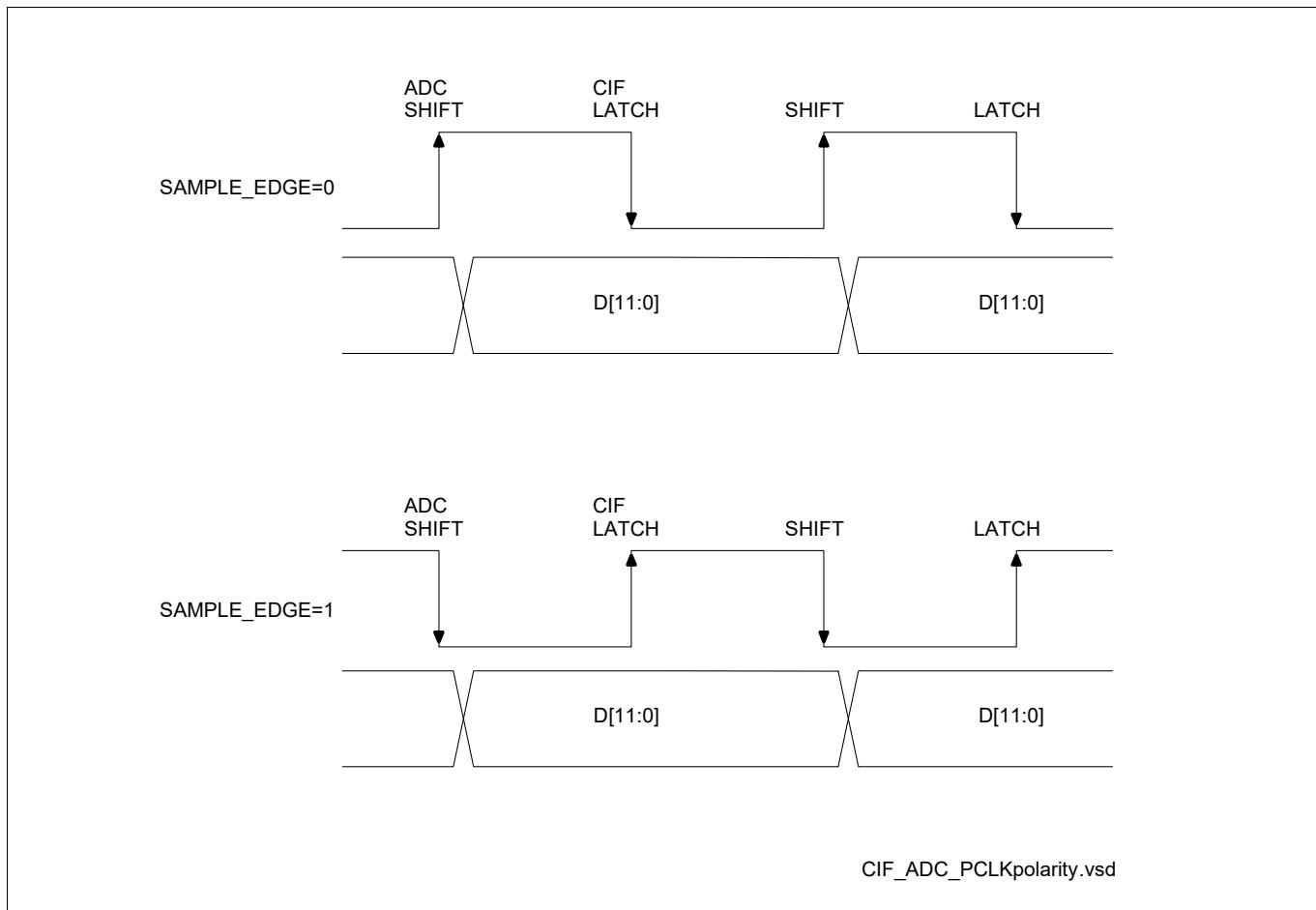
The ADC data has to be connected MSB aligned to the CIF input pins. The [Figure 343](#) shows examples of data connections between a 12-bit or 14-bit ADC and the CIF module.



**Figure 343 External ADC Data Alignment**

## Camera and ADC Interface (CIF)

The pixel clock PCLK controls the data transfer. The first edge of the clock signal can initiate the ADC to provide the data; the second edge can trigger the CIF to latch the data. In the **Figure 344** the providing of the next parallel data is referred to as “shifting”, which is what happens if the ADC has a data FIFO. The clock polarity can be programmed both in the GTM and in the CIF module. In the CIF module, the PCLK polarity is configured in by the bit field **ISP\_ACQ\_PROP.SAMPLE\_EDGE**.



**Figure 344 PCLK Clock Polarity and Data Transfer (12-bit ADC)**

The polarity of the signals HSYNC and VSYNC can also be configured by using the bit fields **ISP\_ACQ\_PROP.HSYNC\_POL** and **ISP\_ACQ\_PROP.VSYNC\_POL**.

## Camera and ADC Interface (CIF)

### 26.3 Functional Description

The CIF forms a complete video and still picture input interface solution for SoCs for various applications.

It is widely programmable and therefore very flexible to use. It supports input frame resolutions up to 4095x4095 pixels.

It consists of a Video Image Signal Processing unit (ISP), a Security Watchdog unit, luminance/chrominance splitter (Y/C Split), a linear downscaling unit, main and several extra cropping units, debug path, JPEG encoder, output unit, control unit (Ctrl) and a BBB slave interface for programming purposes.

All data transfer between EMEM and CIF is handled via the memory interface block.

For more information and a block diagram please refer to the sub module descriptions below and to [Section 26.2.1.1](#).

#### 26.3.1 Sub Module ISP

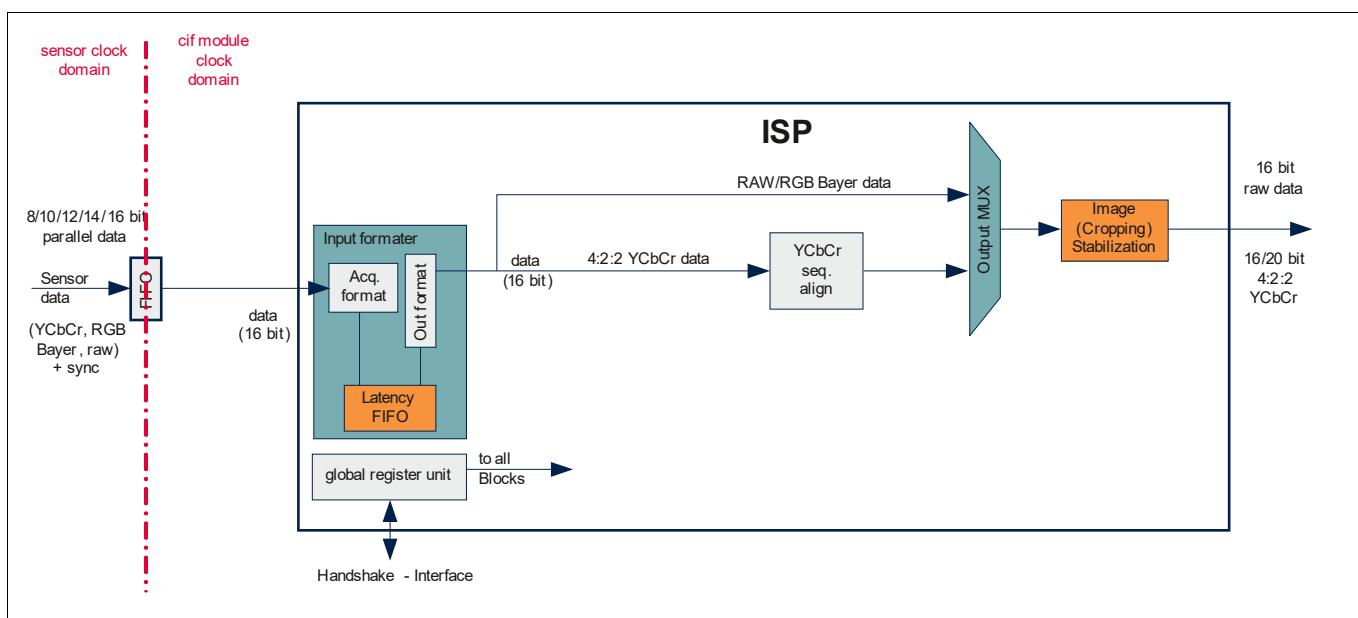
The ISP is the interface to the attached sensor device. It accepts either ITU-R BT.601 YCbCr or RAW RGB Bayer data, or ITU-R BT.656 YCbCr or RAW RGB Bayer data. Furthermore a so called “data mode” is supported which accepts non line or frame organized data. In this mode the states of the hsync and vsync hardware lines are considered as “enable” and “transfer indicator”. This mode allows connecting the CIF to any source delivering a parallel data stream (e.g. camera sensor including a JPEG encoder).

The input part of the ISP is fully programmable in terms of signal polarities, active video data positions, and luminance/chrominance order.

A handshake based mechanism is used for data qualification. As the sensor device can not be stopped from delivering data, a pipeline stall leads to an error if it occurs within an active video line.

The ISP also contains its own programming registers to be accessed by a 32 bit handshake interface.

The ISP can be configured to generate interrupts for multiple different conditions. All interrupts are mapped to the single physical request line ISP\_INT.



**Figure 345 Block Diagram of the ISP Sub Module**

The following features are implemented:

- CCIR656/601 compatible 16-bit video interface
- Input sampling on positive or negative sample clock

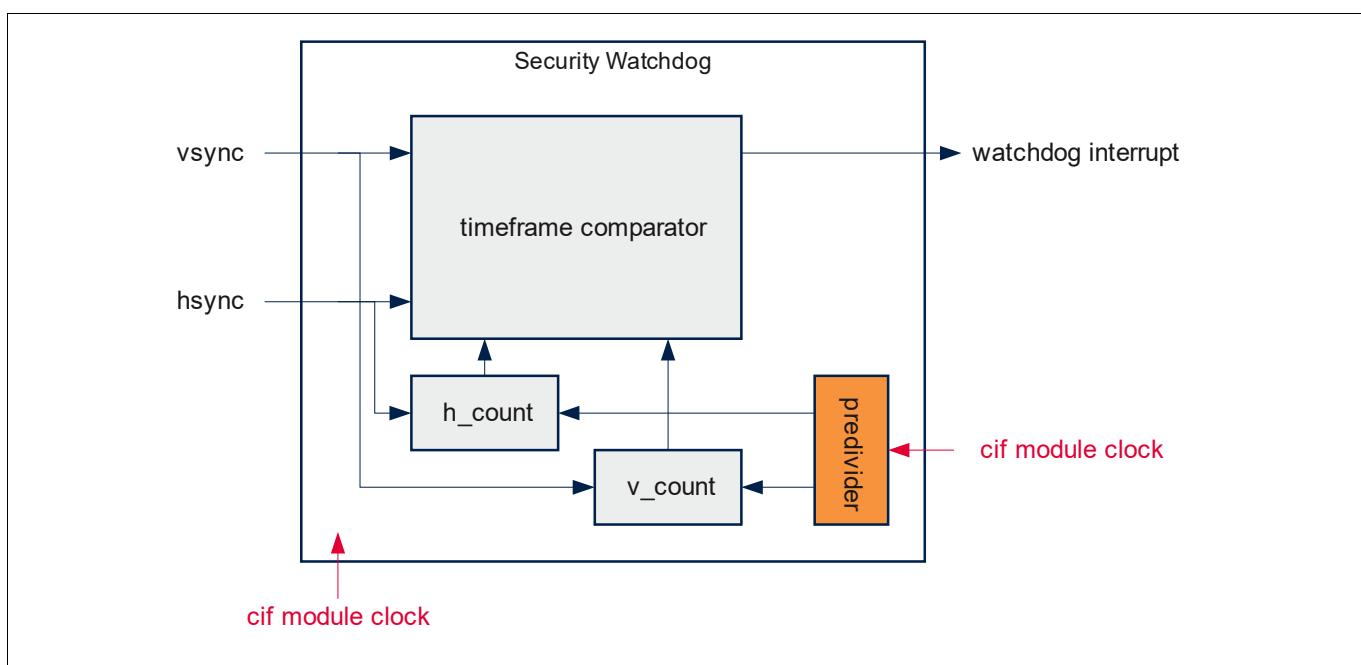
## Camera and ADC Interface (CIF)

- Cropping of the output picture (to crop interpolation artifacts), also used for windowing

### 26.3.2 Sub Module Security Watchdog

The Security Watchdog unit is used to monitor the incoming image data. Specifically it can be used to detect and report irregularities or interruptions in the data stream. To do so the horizontal and vertical synchronization signals in the input formatter unit are observed and are compared to programmable time-out frames. When a time-out frame gets violated an interrupt is generated to immediately report this event (routed to ISP\_INT).

The timing information is retrieved via 16 bit counters. There are two separate counters to measure horizontal and vertical timeframes in parallel. To enhance the configurability of the timing information an additional 16-bit prescaler controls the speed of the counter units, enabling a timing unit granularity from  $1*T_{cif\_clk}$  to  $216*T_{cif\_clk}$ .

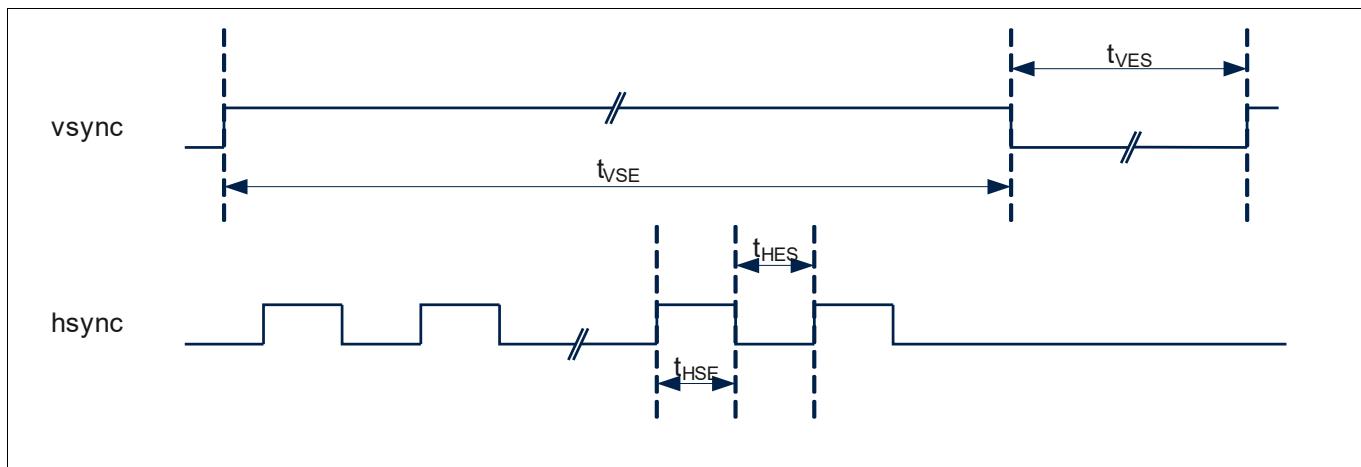


**Figure 346 Block diagram of the Security Watchdog sub module**

As can be seen in [Figure 347](#) there are four time frames of interest to be watched. The time between:

- Start and End of a frame (tVSE)
- End and Start of the next frame (tVES)
- Start and End of a line (tHSE)
- End and Start of the next line (tHES)

## Camera and ADC Interface (CIF)



**Figure 347 Programmable time-outs for the Security Watchdog**

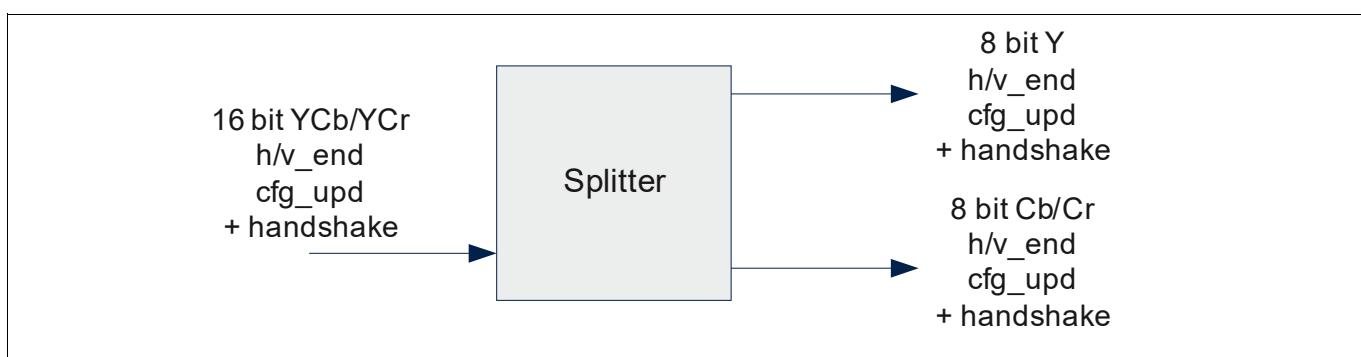
All those time-outs are programmable 16 bit values which can be set via the following registers: “[WD\\_V\\_TIMEOUT](#)” on Page 136 and “[WD\\_H\\_TIMEOUT](#)” on Page 136. So as soon as the corresponding counter reaches the programmed value an interrupt is triggered. A programmed value of zero means the time-out is not configured to trigger an event.

It is possible to completely disable the Security Watchdog unit by setting the corresponding bit of its (“[WD\\_CTRL](#)” on Page 135) or by disabling the clock of this sub module in the “[ICCL](#)” on Page 66.

### 26.3.3 Sub Module Y/C-Split

This module contains a data path splitter separating luminance (Y) and chrominance (C) information of a 16 bit YCbCr 4:2:2. Independent 8 bit Y and C channels are the result of this operation.

The Y/C Splitter must ensure that the output Y and C data streams are fully aligned, i.e. if Y data are processed in one clock cycle only, the C channel has to be stalled until the C component belonging to the processed Y component has been taken. [Figure 348](#) gives a coarse overview of the data flow from/to the sub-module.

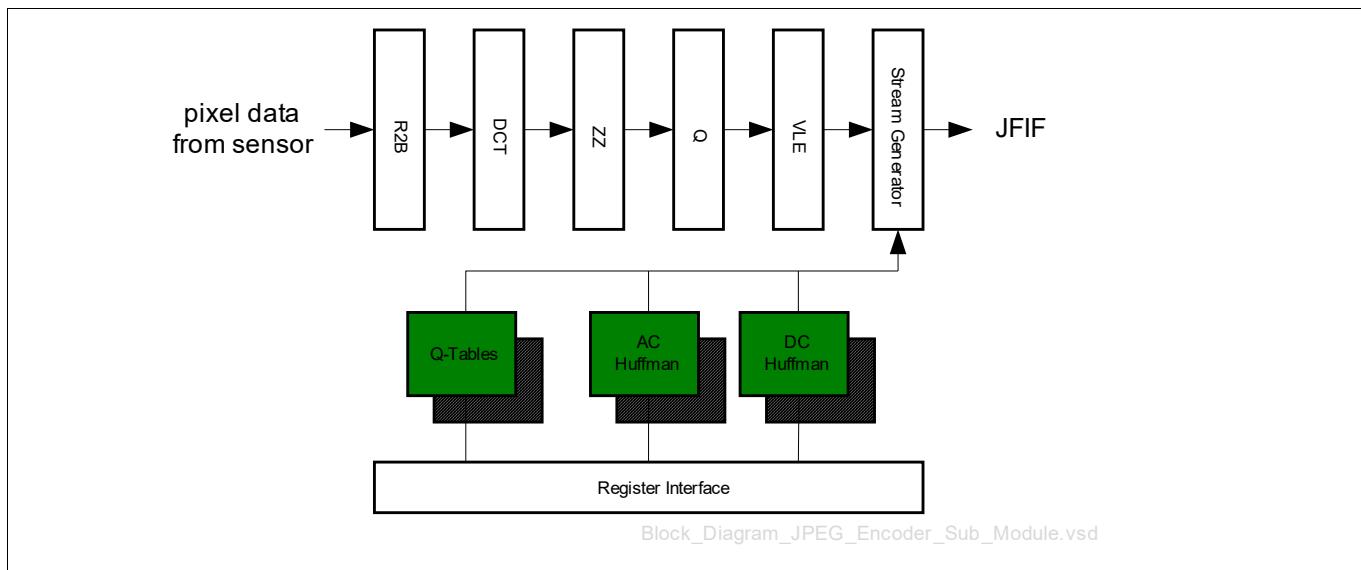


**Figure 348 Block Diagram of the Y/C-Splitting Sub Module**

### 26.3.4 Sub Module JPEG Encoder

The baseline JPEG encoder module consists of a JPEG encoder pipeline. The pipeline - which is shown in [Figure 349 “Block Diagram of the JPEG Encoder Sub Module” on Page 11](#) - is controlled by a register interface that is accessed via handshake interface. The following description gives an overview of the general functionality of the JPEG encoder.

## Camera and ADC Interface (CIF)



**Figure 349 Block Diagram of the JPEG Encoder Sub Module**

The encoding process starts with a raster to block conversion of the  $YC_bC_r$  4:2:2 pixel data provided e.g. by an imaging device. The incoming line oriented pixel data is reordered into 8x8 pixel blocks, one for each component. Every 8x8 pixel block undergoes a baseline DCT computation, a zigzag reordering, a quantization and a variable length encoding (Huffman based). Last encoding step is the generation of the JFIF 1.02 compliant data stream by inserting markers and tables.

The encoder needs two clock cycles per pixel for processing, if  $YC_bC_r$  4:2:2 data is used, which is the normal mode of the JPEG encoder.

The JPEG encoder can be configured to generate interrupts for several error conditions. All interrupts are routed to MJPEG\_INT.

### 26.3.5 Sub Module Linear Downscaler

The first extra path includes a simple scaling unit which allows to reduce the size of the transferred image by simply skipping lines and columns of a frame. The amount of horizontal (columns) and vertical (lines) data reduction can be programmed separately. The module supports four modes:

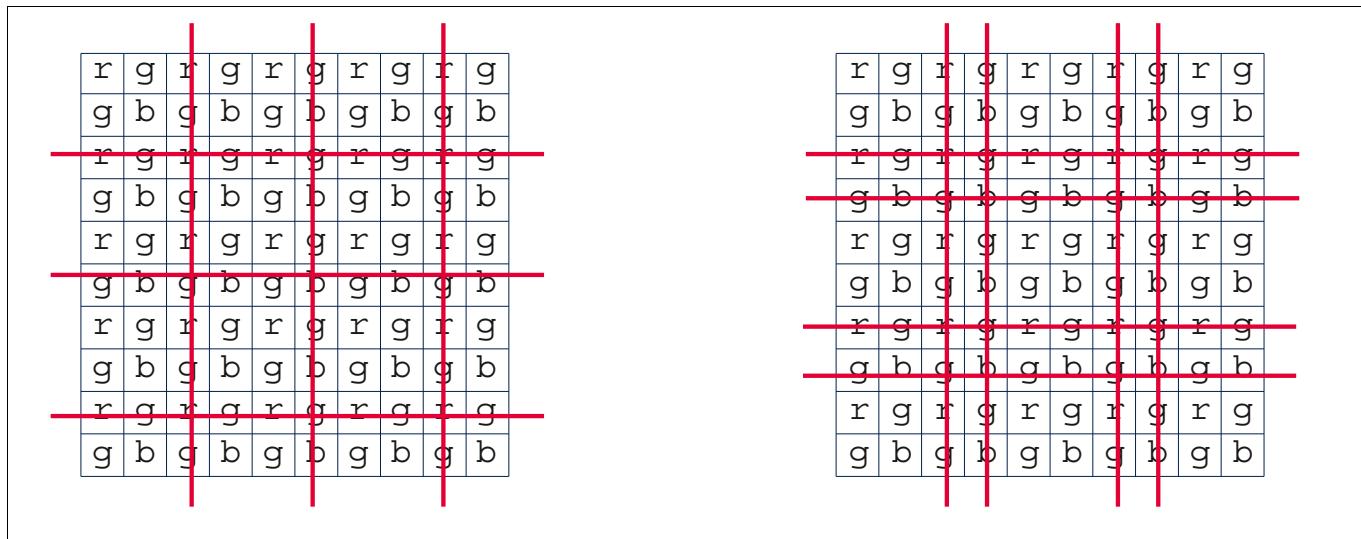
- Single skip
- Double or RGB Bayer skip
- Single pass
- Double pass

The single skip mode simply skips every nth line or column, but since this would destroy the integrity of a regular RGB Bayer pattern - see [Figure 350](#) - there is also the possibility to use the double skip mode which preserves the pattern.

Since skipping leads to a maximum data reduction of 50% at best there are corresponding pass modes to the skip modes. The difference of these modes is that only every nth pixel (or every nth two pixels) get(s) passed to the following processing unit, all other pixels get skipped. In other words: The pass mode can downscale the image data at rates  $\geq 50\%$ , whereas the skip mode allows scaling at rates  $\leq 50\%$ .

Regarding the single/double pass modes it is also important to mention that whether the bayer pattern stays preserved or not is here only dependant on the pass-value. Odd values of n will destroy the bayer pattern integrity, whereas even values will preserve it.

## Camera and ADC Interface (CIF)



**Figure 350 Single vs. RGB Bayer mode**

It's further worth mentioning that the linear downscaler - like any other image processing unit - can be set to a bypass mode where data is passed on unprocessed to the next module.

### 26.3.6 Sub Module Extra Path Units

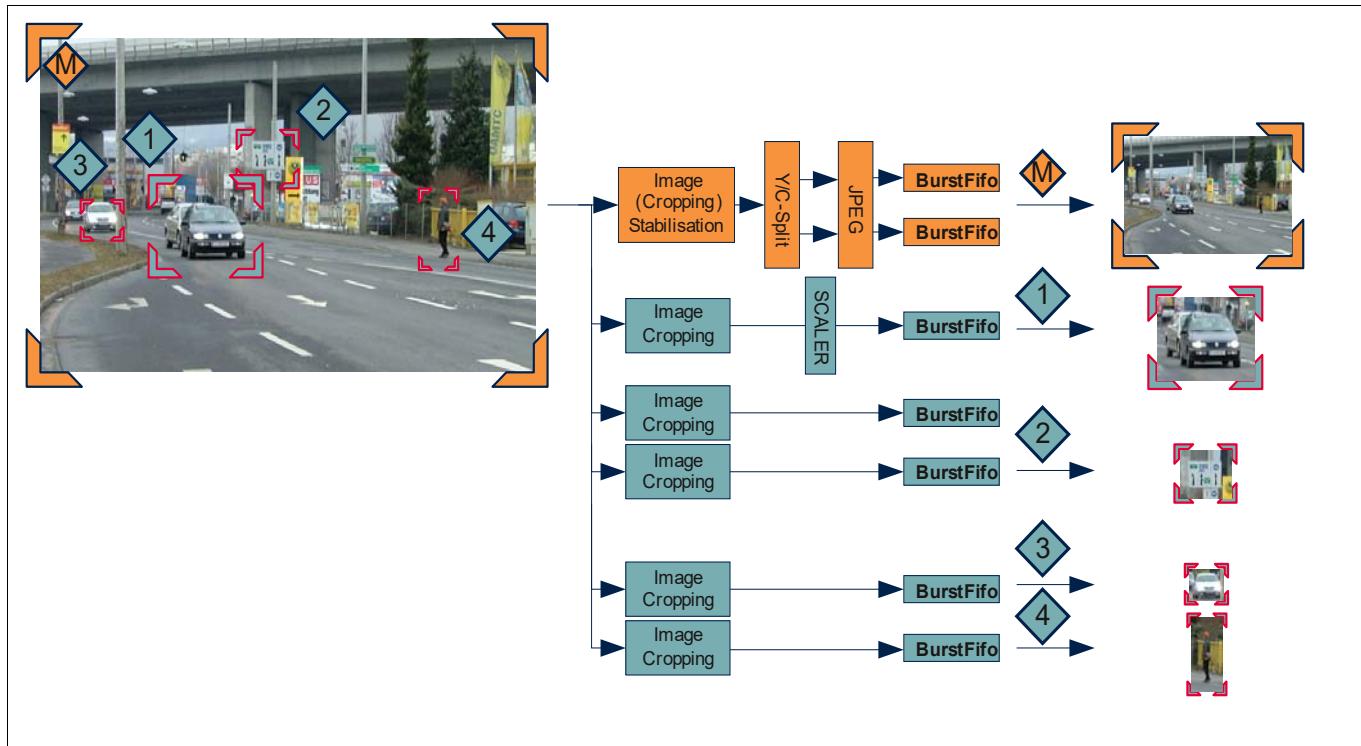
In addition to the main path, the CIF allows definition and transfer of up to 5 additional image regions into memory. Those extra paths all have their own image cropping unit<sup>1)</sup> which allows to crop a separate sub-region of the image. Furthermore it is possible to utilize the displacement and recenter functionality of the image cropping unit to adjust the position of the cropping window (e.g. tracking a moving object). For details about the functionality of the image stabilization unit see the corresponding section in the CIF Programmer's Manual.

**Figure 351** shows the function of the main and extra path units: The main path may capture the full incoming frame utilizing it's downscaling unit to keep the output size - and hence the computational effort for ongoing image processing by the system - low.

1) In the main path the Image Cropping unit is referred to as Image Stabilization unit, since in YCbCr video mode it may be used to compensate camera shakes.

## Camera and ADC Interface (CIF)

In parallel up to 5 (the example below shows four) sub images may be cropped from the same input image. Those images are stored and handled independent of the main path and also independent of each other.



**Figure 351 Example: Main and four active Extra Paths**

Since the Y/C-Split unit is not present in the extra paths, the following restriction is given:

- only one receive buffer (no support for planar/semi-planar storage)

Nevertheless every extra path has its own storage ring buffer and its own distinct interrupt generation unit for those. Those separate interrupts of the distinct extra paths are then again merged in a single multiple source interrupt register to a single separate extra path interrupt line (MIEP\_INT).

Only the first extra path has a downscaling unit, so for the other paths (including the main path) it is not possible to reduce the image dimension further after the cropping stages.

Every present extra path can be en/disabled via a programmable control register.

### Parallel use of several extra paths

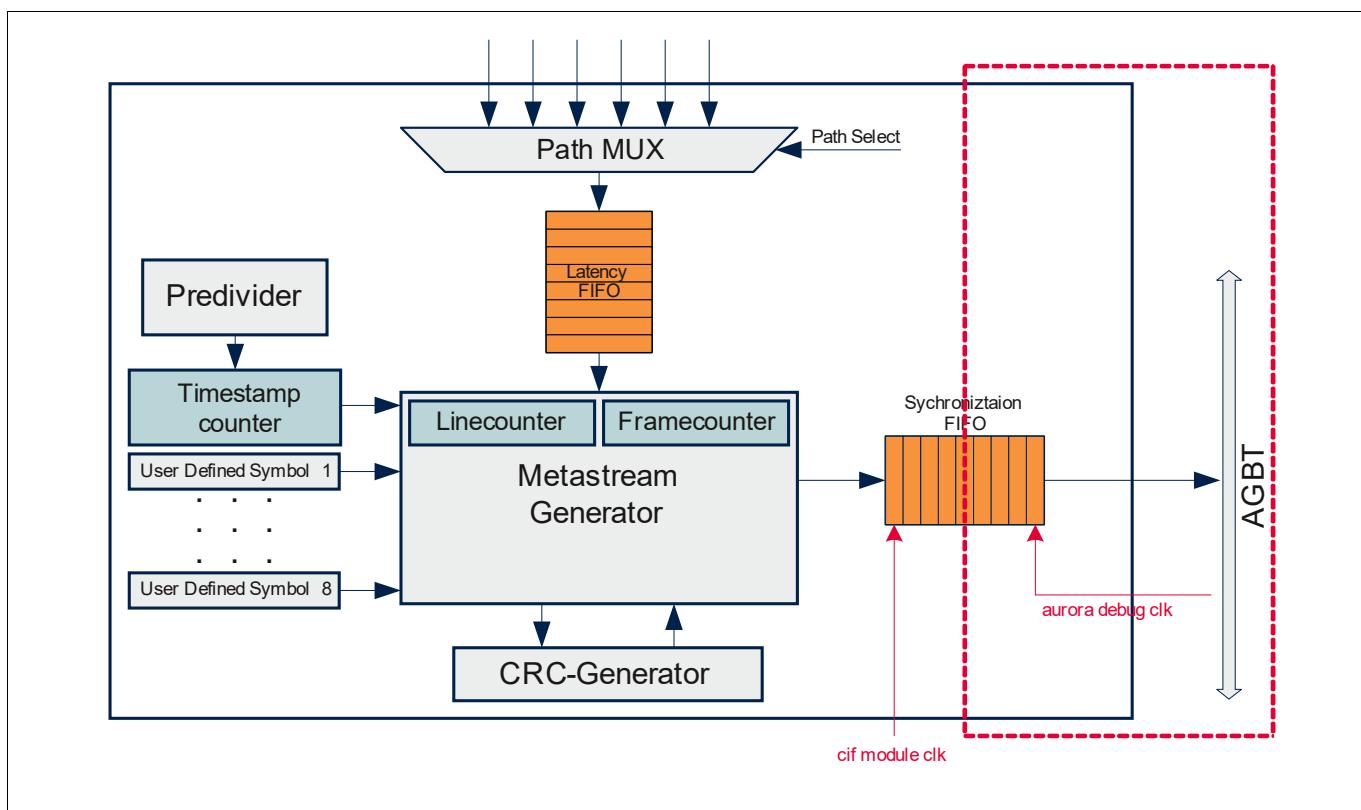
Using multiple overlapping or nested cropping windows leads to parallel activation of several extra paths generating bus load that could overload the memory interface and the back bone bus in case high baud rates are used. Use of more than two nested windows is not recommended. Extra care should be taken when using overlapped cropping windows to keep the peak BBB bus load low. For example, use cropping windows widths which are multiple of the burst size configured in the MI (Memory Interface) block. This allows the cropped regions to be transferred to memory using only bursts on the BBB bus, without a need for single moves to transfer the pixels at the end of a line that do not make a whole burst. For burst of four 32-bit moves, resulting in 16 bytes per burst, the optimal cropping window width is multiple of 16 pixels (1 byte per pixel). CIF single move takes 8 BBB clock cycles (0.5 bytes/clock cycle), burst of four 12 (1.33 bytes/clock cycle), and burst of eight 16 clock cycles (2 bytes/clock cycle). Burst of eight results in 200 Mbytes/sec for 100MHz BBB bus clock, which is the theoretical maximum if the CIF module is the only master on the BBB bus.

## Camera and ADC Interface (CIF)

### 26.3.7 Debug Path

The debug path is used to transfer data from one of the CIF paths (main or extra) to an Aurora Gigabit Trace (AGBT) debug interface. To do so the data from one of the paths - which may also still be passed on to the memory interface - is additionally routed to the debug unit. Despite simply passing this data on to the AGBT interface, the CIF debug path also inserts meta information and CRC checksums to the stream. This helps to ease, enhance and secure the analysis of the retrieved data.

As can be seen in the **Figure 352** it is possible to select - via a MUX - from which path the data to send via the Aurora interface is taken. The selected data is then transferred into a latency compensation FIFO. This is needed to let the Metastream Generator send several additional information symbols before the received data stream, which shall not block the transfer of this stream to memory meanwhile.



**Figure 352** Debug Path block diagram

Those additionally inserted symbols are:

- Metasymbol marker
- Metastream type information
- a 30-bit timestamp
- a Frame-/Linenumber
- up to eight 15-bit user defined symbols
- Payload length Information
- Header & Payload checksum

**Figure 353** shows an example of a pixel line as transferred over the debug interface. The pixel debug stream consists of 16-bit symbols, where the value  $FFFF_H$  shall be a reserved value - the so called Metasymbol marker - which distinctly marks the start of a debug frame. This leads to some obvious restrictions in other symbols. The general rule reads as follows: In every symbol despite the Metasymbol marker the highest bit (MSB) has to be  $0_B$ .

## Camera and ADC Interface (CIF)

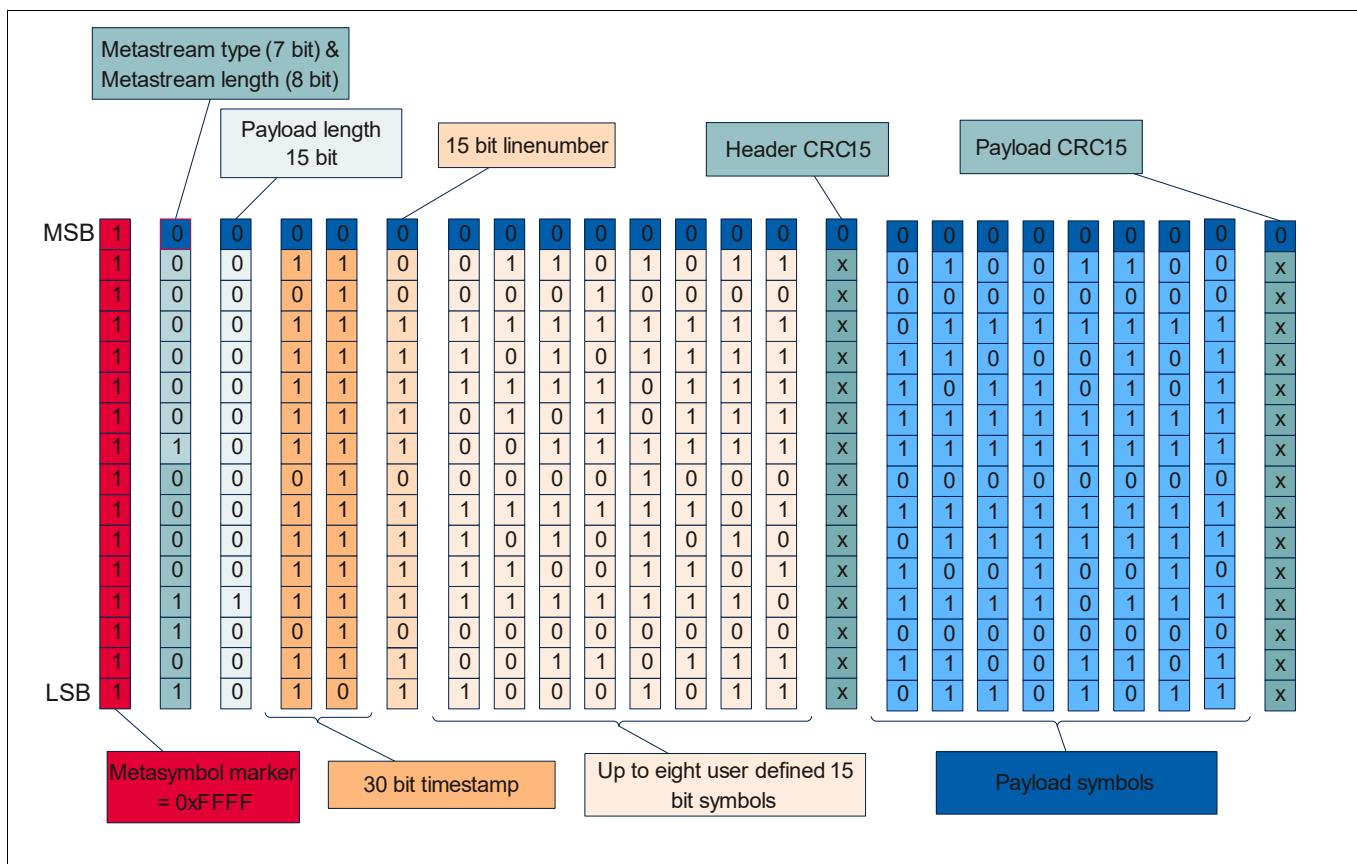
The currently supported Metastream types are:

- Start of Frame (SoF) Code  $00_H$
- Start of Line (SoL) Code  $01_H$

SoF is a simple Marker which only consists of the type and length information, timestamp, framenumber, user symbols and CRC. The SoF has no payload attached and hence no corresponding information like payload length or payload-CRC. Like the name suggests the SoF is sent only once at the beginning of an image frame. It has to be followed by - depending on the image size - several SoL packets.

**Figure 353** shows an example of a SoL packet. Basically it contains the same information like a SoF packet, just with a linenumber instead of a framenumber and additional payload information.

As can be also seen in the example the payload symbols - which are the image pixels of a line - are right shifted by one & stuffed with zero on the high significant end. In case of 16 bit data this means that the least significant bit of the payload gets lost. In 14 bit data mode, the data is right-shifted for 1 bit, but the two LSB bits are always zero, so there is no data loss. In YUV16 mode the module shifts right and the chrominance LSB is lost. In RAW16 mode the module does not shift and the LSB bit is lost.



**Figure 353** Line debug packet

The header length which is always included in a packet and the payload length which is additionally included in SoL packets can be used as additional security measures to quickly identify transmission gaps or similar issues. All given length information in the debug stream is referring to symbols. So in the example of **Figure 353** the payload length of 8 means that exactly 8 symbols of payload have to follow the header. The header length symbol gives the length of the header excluding the Metasymbol marker. For both values, header and payload - the CRC is not included.

The timestamp is a simple free running 30 bit counter and can be used to introduce quite accurate timing information to the stream. To enhance the flexibility of this counter a preceding prescaler unit is included which

## Camera and ADC Interface (CIF)

is clocked by the CIF module clock. This unit allows to configure the granularity of the timestamp to be between 1\*Tcif module clk to 232\*Tcif module clk.

User defined symbols are a way to insert/feedback software dependant information into the data stream. One possible application could be the insertion of coarse time/date information which supplements the internally generated timestamp.

The eight user defined symbols are freely programmable registers which can be included into the stream on a per symbol basis. Per default after reset no user defined symbols will be sent in the debug data stream.

Linenumber and framenumber are simple counters which can be reset by software at any time. The framenumber will run free and simply overflow starting again from zero, whereas the linenumber will be reset internally with every new SoF packet.

As CRC a 15 bit version using the same polynomial as the CAN-bus CRC is used. The input to this CRC is a full 16 bit symbol. The header CRC covers all symbols from the Metasymbol marker up to the last active user defined symbol. Whereas the payload CRC covers all payload symbols from the first after the header CRC up to the last.

If it is not needed it is possible to deactivate the whole debug path by disabling it's clock via the "["ICCL" on Page 66.](#)

**Note:** *The CIF module shifts the pixel data to the right when transferring via AGBT, in order to guarantee data symbol MSB of zero. Therefore, the LSB bit is lost in case the data is LSB aligned. This is valid for YUV data and 16-bit radar data. The raw RGB data is 14-bit MSB aligned and no LSB loss occurs.*

### Activating and Deactivating the AGBT Path

When activating the AGBT path, use the following sequence:

- first select CIF in the AGBT module then configure the AGBT registers in CIF and at the end enable the AGBT path in the CIF module

When deactivating the AGBT path, use the inverse sequence:

- first disable the AGBT path in the CIF module then deselect CIF in the AGBT module

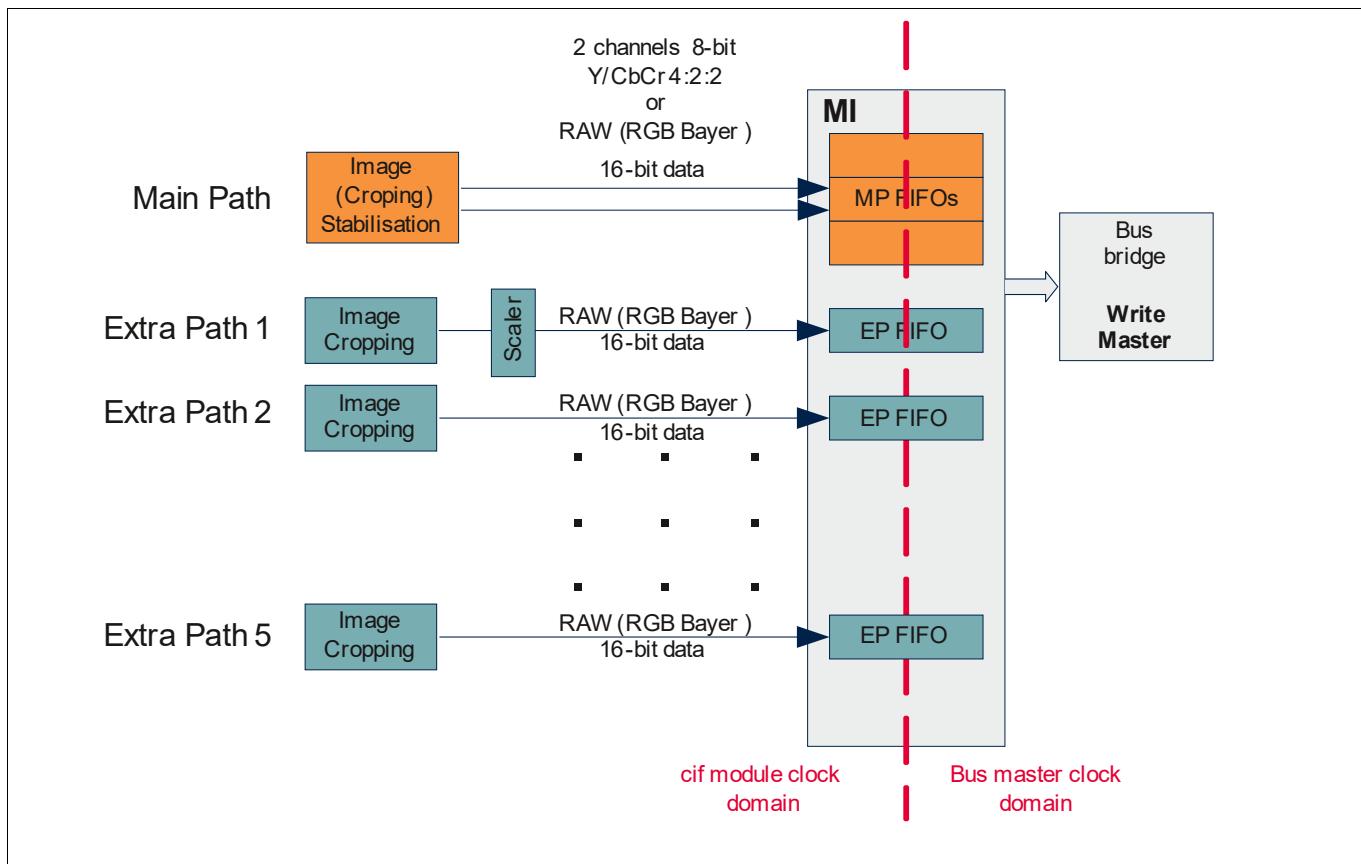
### 26.3.8 Sub Module Memory Interface (MI)

The Memory Interface module - shown as in [Figure 354](#) - has the following tasks:

- It collects the CIF internal data streams and writes them into the EMEM.
- It decouples the BBB master clock from the CIF module clock.

**Attention:** *In order to configure the CIF BBB master to access the EMEM, please refer to the system memory map.*

## Camera and ADC Interface (CIF)



**Figure 354 Overview Memory Interface connections**

The MI supports storage of main picture YCbCr, JPEG and RAW data (up to 16 bit). Y, Cb and Cr data is stored in EMEM supporting the following modes:

- separate buffers (planar)
- two buffers (semi planar)
- one buffer (interleaved)

**Figure 355** gives an overview of those three modes, but also shows the effect of the byte swapping option which effects the endianness of the output format.

All FIFOs used in MI are synchronization FIFOs which handle the task of decoupling the different clock domains.

## Camera and ADC Interface (CIF)

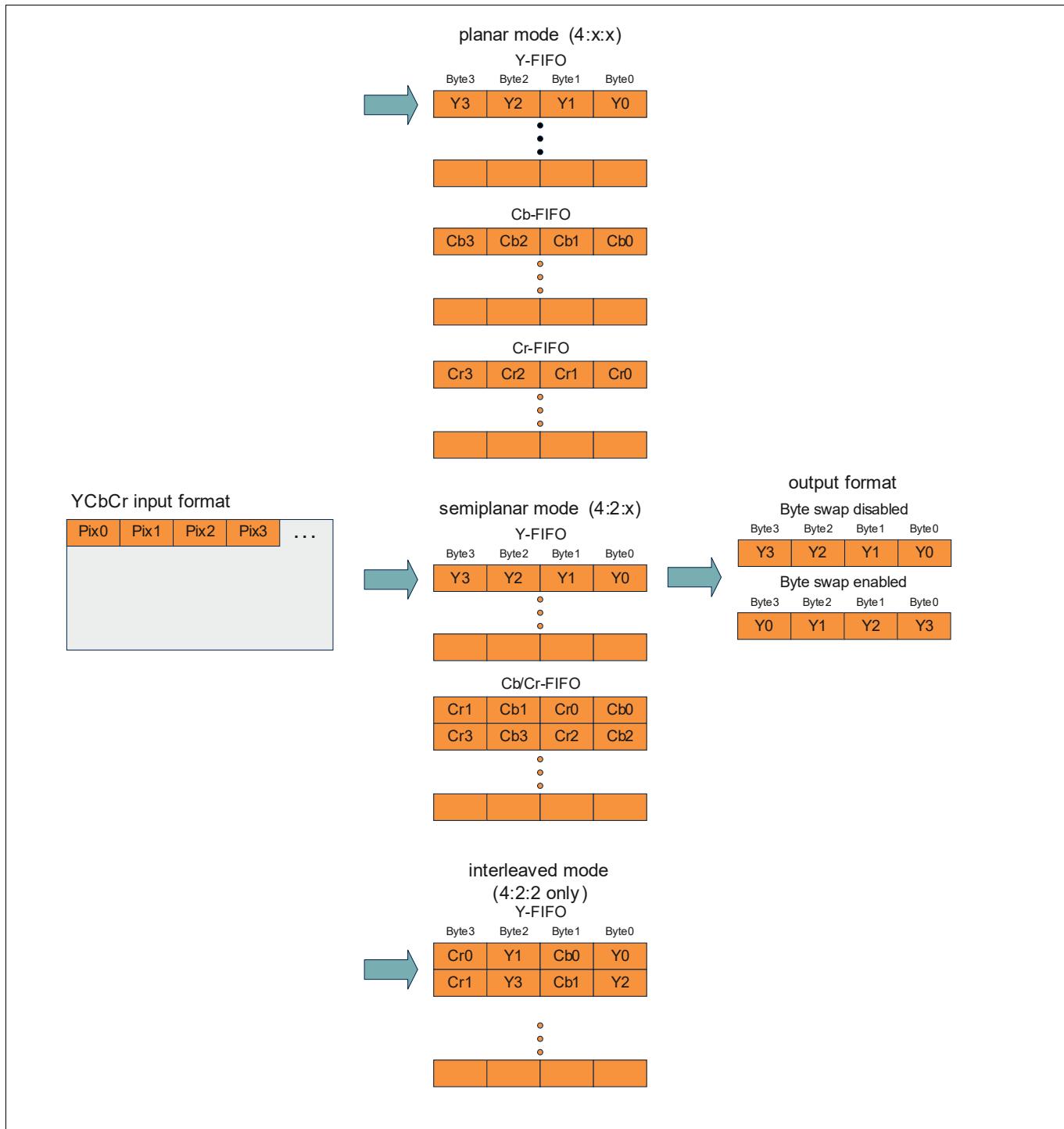
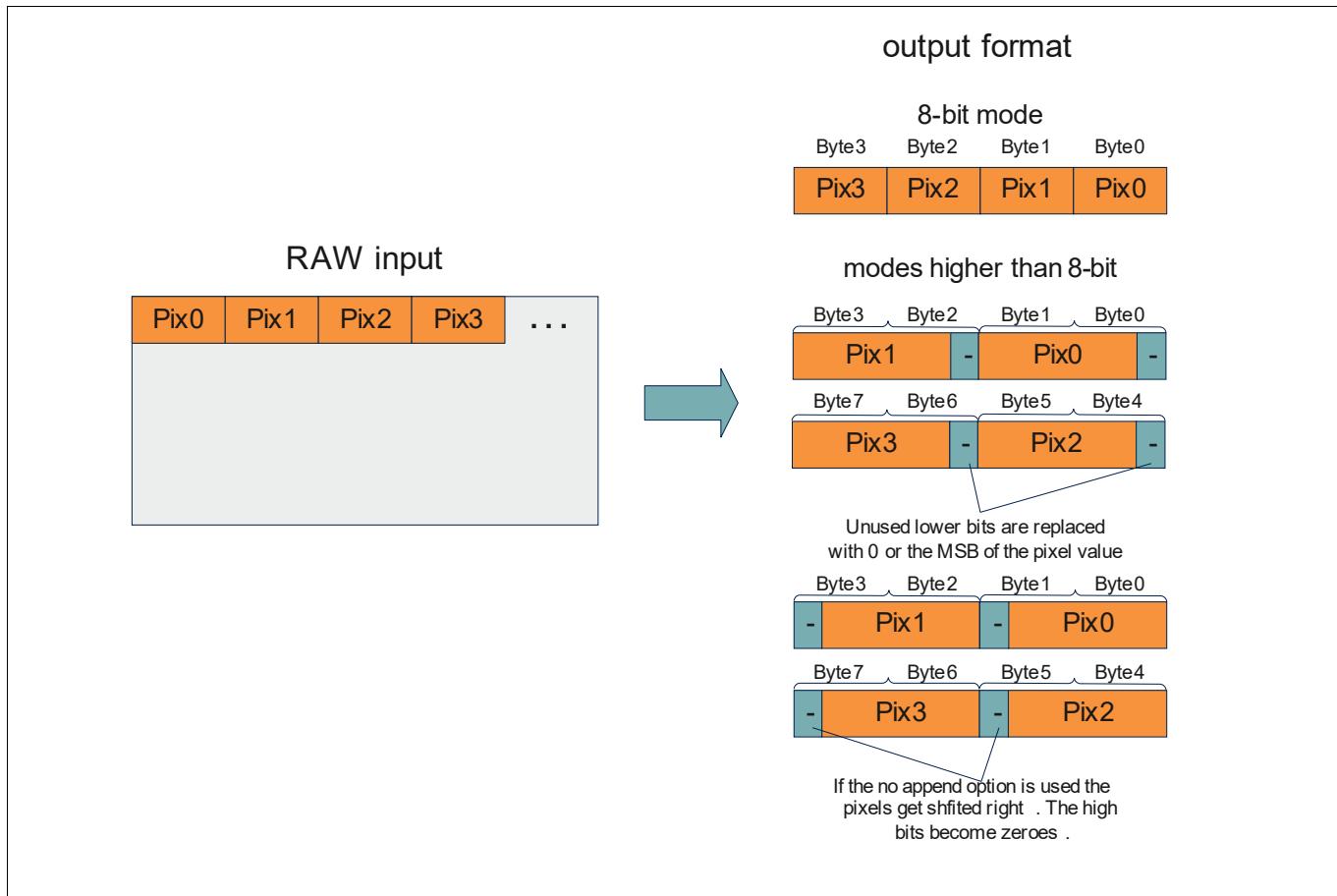


Figure 355 Storage scheme YCbCr in MI

For storage of RAW data only one FIFO buffer is supported/needed. If more than 8 bits data are required in RAW mode, then the data bits are stored as 16-bit words in memory. Usually these bits are stored MSB aligned, meaning the unused least significant bits remain zero or become a replica of the RAW data most significant bits (see “[INPUT\\_SELECTION](#)” on [Page 72](#)). It is also possible to disable this MSB alignment (“[INPUT\\_SELECTION\\_NO\\_APP](#)” on [Page 72](#)) leading to LSB aligned data values. This especially makes sense for non image RAW data (e.g. data mode). [Figure 356](#) visualizes the previously mentioned facts.

## Camera and ADC Interface (CIF)



**Figure 356 RAW data storage scheme in MI**

Unlike the main path the extra paths support interleaved storage mode only. This is given by the fact that extra paths only have a single component FIFO.

### 26.3.8.1 Write to EMEM

The following [Figure 357](#) shows the basic building blocks of the unit. As can be seen there are several input port handshake interfaces:

- 2 x 8 bit Y, Cb/Cr main picture data or up to 16 bit RAW data
- 64 bit JPEG data
- For every extra path 2 x 8 bit Y, Cb/Cr extra picture data<sup>1)</sup> or up to 16 bit RAW data

The following modes of operation are supported:

- image data only
- JPEG data only
- RAW data only

The interface to the EMEM is a simple handshake oriented protocol, which is connected to the BBB master via a bridge. Three kinds of bus accesses are supported, 8-beat bursts, 4-beat bursts and single transfers, all with a data width of 32 bit. The storage areas, which are located in the EMEM, are organized as ring buffers.

Interrupts are generated for the following events:

- picture end of frame

1) YCbCr data is handled interleaved as 16 bit data extra paths.

## Camera and ADC Interface (CIF)

- macro block line (16 lines of Y data and 8 lines of Cb and Cr data of the picture are written to the EMEM)
- programmed fill level of picture Y data reached
- wrap around of the programmed address range separately for Y, Cb and Cr of the picture
- BBB write error propagated by the handshake target interface

Interrupts are mapped onto a single, physical request line MI\_INT for the main path and interrupts associated with the extra paths are mapped to MIEP\_INT.

Each FIFO can store at least two 8-beat bursts.

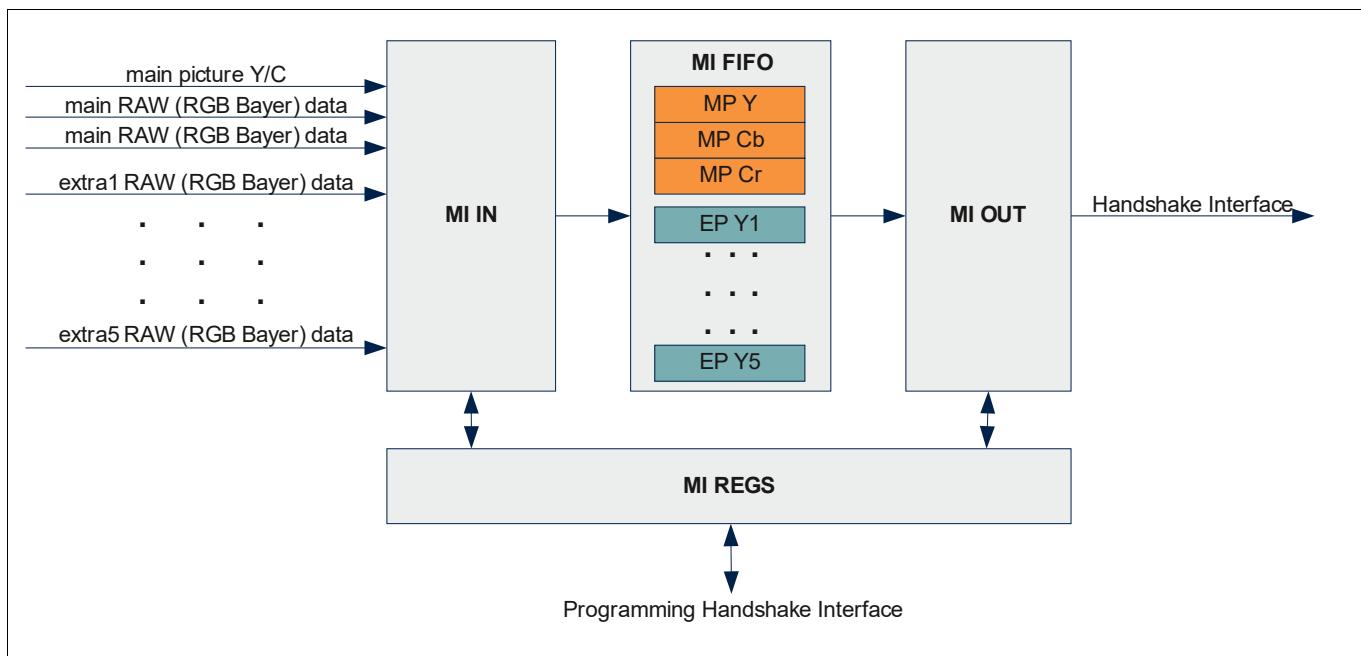


Figure 357 Block Diagram of the Memory Interface write path

### 26.3.9 BBB Master Interface

The BBB Master Interface connects the Memory Interface to the BBB.

- Unique TAG
- Word Access only
- Priority relative to CPU access via EMEM
- Bandwidth on bus
- RAM arbitration inside EMEM
- Reset behavior
- Bus error handling

## Camera and ADC Interface (CIF)

### 26.3.10 BBB Slave Interface

The BBB Slave Interface connects the Control Unit to the BBB system. Internally it represents an AHB-Slave-to-handshake-Initiator block, while from the BBB a standard SPB\_BPI is used.

- No safety features.
- No OCDS functionality.
- No ENDINIT/OEN/... protection
- Standard CLC register.
- Allowed access width is 32-bit only.  
8-bit and 16-bit accesses are treated as 32-bit and the content of all 32 data signals of the bus is written to the register.
- Error handling - no notification of read and write accesses to reserved locations

### 26.3.11 Control Unit

The Control Unit serves two purposes:

- Interface from BBB Slave Interface to the local configuration register blocks of the other modules
- Clock and reset control registers for the CIF

The Control Unit has one 32 bit handshake input interface, three 16 bit handshake output interfaces, and two 32 bit handshake output interfaces.

As all transfers from the handshake input interface are 32 bit wide, these accesses have to be mapped accordingly to the 16 bit handshake interfaces, e.g. ignoring the upper 16 bits or filling them with zeros respectively.

Each block inside the CIF uses a dedicated clock signal that can be controlled by a programmable register inside the Control Unit.

Existing software resets in the CIF blocks can also be controlled by a programmable register inside the Control Unit. An asynchronous reset for the sensor clock domain has to be generated from the system reset. A soft reset for all registers in the CIF is provided. It behaves like the asynchronous hardware reset.

**Attention:** *When resetting the whole CIF module, use the [BBB\\_KRST0](#) register for the whole module, and not the partial resets from the register [IRCL](#).*

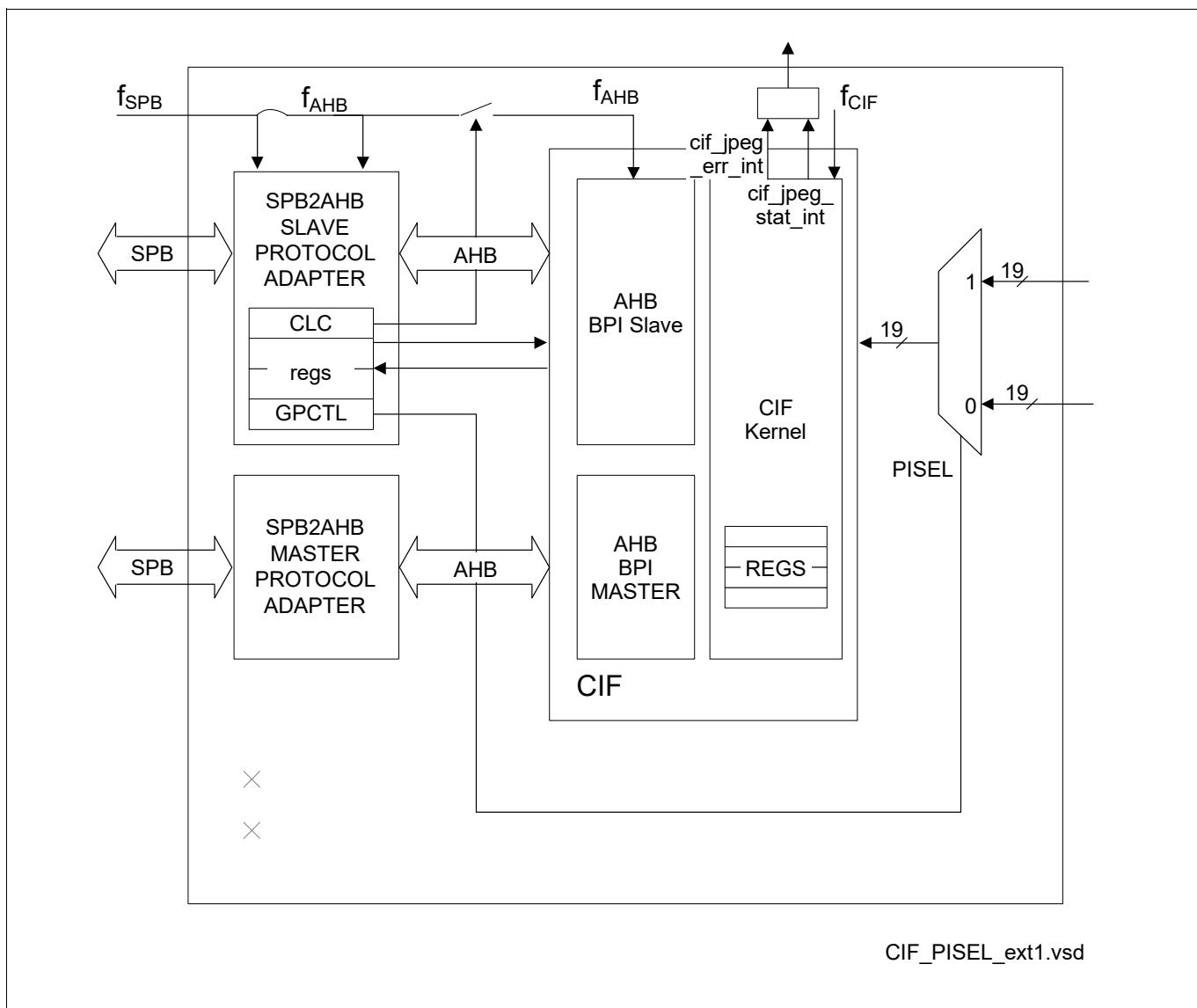
### 26.3.12 Shadow Registers

A distinctive feature of the local configuration registers is the implementation of “shadow registers”. Because it can not be ensured that the complete re-programming of the CIF can be done during the vertical blanking period between two frames shadow registers have been introduced. These shadow registers hold the configuration values that are actually used for processing and are only updated with the latest written configuration values after a special update signal is triggered. This update signal may be generated after a frame has ended but in order to allow a fast configuration it may also be forced to trigger immediately.

## Camera and ADC Interface (CIF)

### 26.3.13 CIF Module Integration and BPI Adapter

This section describes the implementation and integration details of the CIF module.



**Figure 358 Overview of the SPB2AHB adapter**

**Note:** Normally only one pin mapping of CIF signals to pins is available, selected by **BBB\_GPCTL**. PISEL default value of 0. The second selection is tied to low.

#### 26.3.13.1 BPI\_SPB Module Registers

This section describes the registers implemented in the slave (SPB2AHB) adapter component.

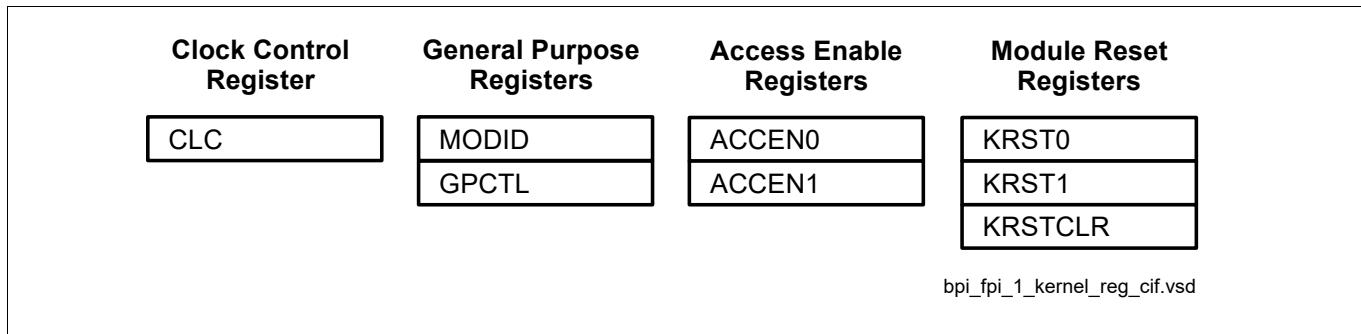
In this specification release it is assumed that the BPI\_SPB registers will use reserved addresses in the module address space, and not separate address page. This issue is subject of discussion and may change.

## Camera and ADC Interface (CIF)

### 26.3.13.1.1 System Registers

**Figure 359** shows all registers associated with the BPI\_SPB module, configured for one kernel. The Offset Address in the following BPI\_SPB register table and in the BPI\_SPB register descriptions are proposals. In a standard 32 bit peripheral the CLC is mapped to offset address 00h, module ID to 08h.

#### BPI\_SPB Registers Overview



**Figure 359 BPI\_SPB Registers**

#### Standard Registers

The adapter will implement the following standard registers as defined in the specification for the SPB BPI:

- MODID, Module ID<sup>1)</sup>
- CLC, Clock Control
- ACCEN0, Access Enable 0
- ACCEN1, Access Enable 1
- KRST0, Kernel Reset 0
- KRST1, Kernel Reset 1
- KRSTCLR, Kernel Reset Clear
- GPCTL Register

Instantiation of the CLC register is controlled by a VHDL generic. Instantiation of the KRST0, KRST1, and KRSTCLR registers is controlled by a second VHDL generic. The MODID, ACCEN0 and ACCEN1 registers will always be present.

If a particular register is not instantiated then any associated output ports on the adapter entity will be driven to a constant logic value.

#### Kernel Reset

If a kernel reset is requested, the adapter will synchronously assert the internal kernel reset output and error any ongoing accesses on the SPB bus not addressed to the adapter's internal SFRs.

The method of using the kernel reset output from the adapter to initialise the associated AHB module is module specific and outside the scope of this specification.

Any AHB accesses in progress will be synchronously terminated.

1) It is possible to set the fields of the module ID register without altering the adapter RTL by setting VHDL generics on the adapter instantiation

## Camera and ADC Interface (CIF)

### GPCTL

There is a single adapter specific SFR implemented in the adaptor.

The SFR contains 32 general purpose read/write control bits without ENDINIT protection. All the bits are connected to output ports on the SPB2AHB entity. The SFR will implement no safety requirements and is not suitable for controlling hardware related to safety functions. The SFR supports, byte, half word and word transactions only. All other transactions are rejected with an SPB error termination.

*Note:* In the CIF module, this register is reserved.

### Principle of Operation

When the controlling state machine detects an SPB access, it compares the address to the address of the SFR. This address is configurable via VHDL generic when instantiating the adaptor. In the case that the address matches the SFR address, the normal state machine transitions are interrupted and the access is not passed to the AHB bus.

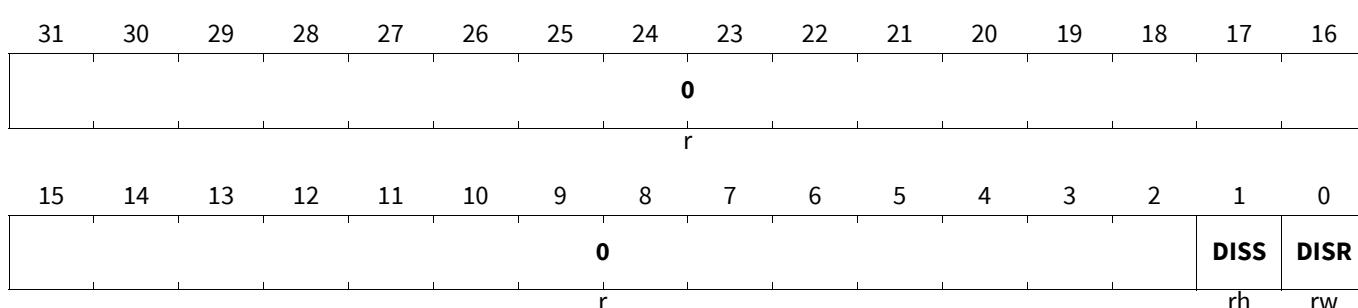
If the access opcode is not SDTB, SDTH or SDTW, then the access is terminated with an SPB error. If the opcode is supported, the SFR data will be returned on the SPB bus if the transaction is a read or the appropriate bits of the register will be updated if the access is a write.

### Clock Control Register

The Clock Control Register, CLC, acts globally and allows the complete AHB module to be disabled to reduce power consumption when the module is not required. When the module is disabled ( $\text{DISS}=1_B$ ), only register accesses to the adapter register address space are permitted. All other accesses to the module address space will be errored.

#### BBB\_CLC

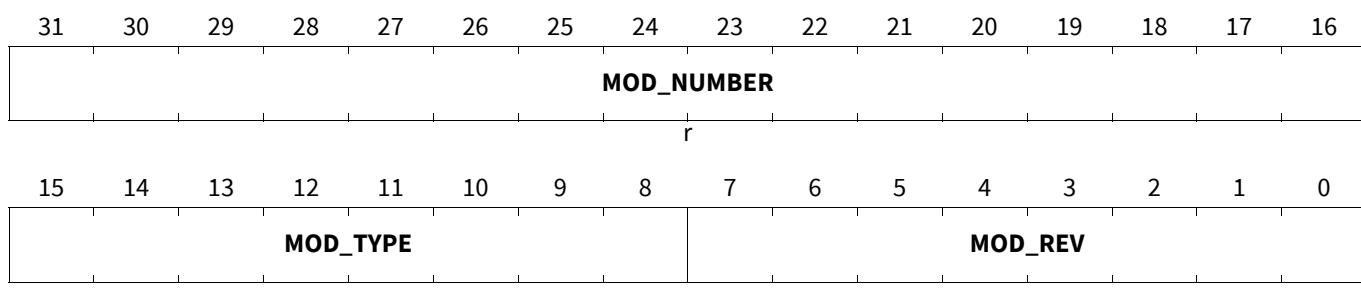
**Clock Control Register** (0000<sub>H</sub>) Application Reset Value: 0000 0003<sub>H</sub>



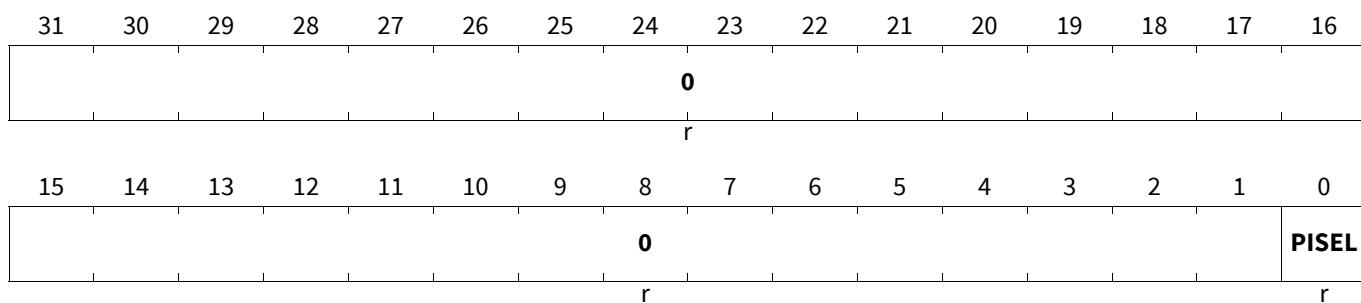
Field	Bits	Type	Description
<b>DISR</b>	0	rw	<b>Module Disable Request Bit</b> Used for enable/disable control of the module.
<b>DISS</b>	1	rh	<b>Module Disable Status Bit</b> Bit indicates the current status of the module.
<b>0</b>	31:2	r	<b>Reserved</b> Read as 0; should be written with 0.

### Module Identification Register

The identification register allows the programmer version-tracking of the module. The table below shows the identification register which is implemented in the DMA module.

**Camera and ADC Interface (CIF)****BBB\_MODID****Module Identification Register**(0004<sub>H</sub>)Application Reset Value: 00B3 C0XX<sub>H</sub>

Field	Bits	Type	Description
MOD_REV	7:0	r	<b>Module Revision Number</b> This bit field defines the module revision number. The value of a module revision starts with 01H (first revision).
MOD_TYPE	15:8	r	<b>Module Type</b> The bit field is set to C0H which defines the module as a 32-bit module.
MOD_NUMBE R	31:16	r	<b>Module Number Value</b> This bit field defines a module identification number.

**General Purpose Control Register****BBB\_GPCTL****General Purpose Control Register**(0008<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
PISEL	0	r	<b>Port Input Select Bit</b> Selects between two pin mappings of the CIF interface. 0 <sub>B</sub> pin mapping 0 selected (default) 1 <sub>B</sub> pin mapping 1 selected
0	31:1	r	<b>reserved</b> Reads 0. Should be written with 0.

**Access Enable Register 0**

The Access Enable Register 0 controls write access for transactions to registers with the on chip bus master TAG ID 000000<sub>B</sub> to 011111<sub>B</sub> (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The

## Camera and ADC Interface (CIF)

adapter is prepared for a 6 bit TAG ID. The registers ACCEN0 / ACCEN1 provide one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ..., EN31 -> TAG ID 011111B.

### **BBB\_ACCENO**

#### **Access Enable Register 0 (000C<sub>H</sub>) Application Reset Value: FFFF FFFF<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rw															

Field	Bits	Type	Description
ENy (y=0-31)	y	rw	<b>Access Enable for Master TAG ID y</b> This bit enables write access to the module register addresses for transactions with the Master TAG ID y 0 <sub>B</sub> Write access will not be executed. Read accesses will be executed. 1 <sub>B</sub> Write and read accesses will be executed

### **Access Enable Register 1**

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). These tags are not used in this system and so no programmable bits are provided.

### **BBB\_ACCEN1**

#### **Access Enable Register 1 (0010<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES															
r															

Field	Bits	Type	Description
RES	31:0	r	<b>Reserved</b> Read as 0; should be written with 0.

### **Kernel Reset Register 0**

The Kernel Reset function is used to synchronously reset the AHB module. To activate the kernel reset, it is necessary to set the RST bits by writing with 1<sub>B</sub> in both Kernel Reset Registers. The RST bit will be re-set by the adapter with the end of the adapter kernel reset sequence.

## Camera and ADC Interface (CIF)

Kernel Reset Register 0 includes a kernel reset status bit that is set to  $1_B$  in the same clock cycle the RST bit is reset. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLE.CLR register bit.

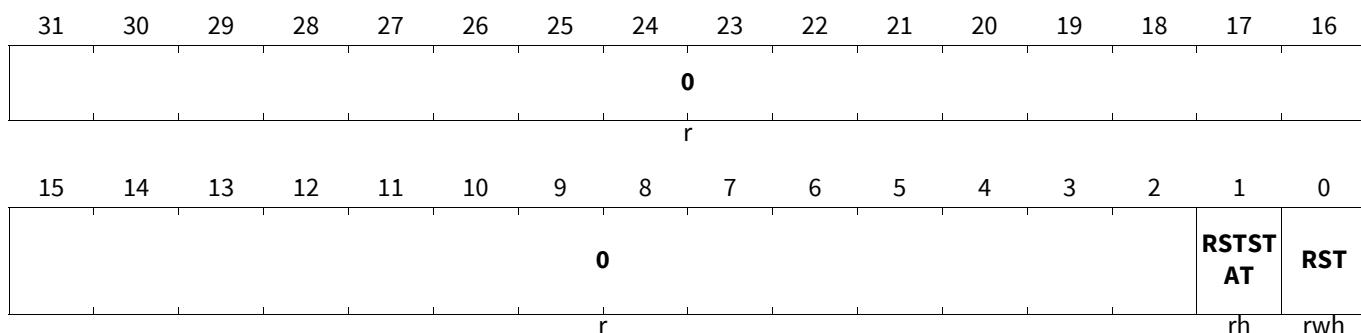
During the execution of the kernel reset until RSTSTAT is set, write accesses to the module registers will result in an error acknowledge. Adapter registers can still be accessed.

### **BBB\_KRST0**

#### **Kernel Reset Register 0**

(0014<sub>H</sub>)

**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>RST</b>	0	rwh	<b>Kernel Reset</b> This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to $0_B$ ) after the kernel reset was executed. $0_B$ No kernel reset was requested $1_B$ A kernel reset was requested
<b>RSTSTAT</b>	1	rh	<b>Kernel Reset Status</b> This bit indicates whether a kernel reset was executed or not. This bit is set after the execution of a kernel reset in the same clock cycle both reset bits. This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register. $0_B$ No kernel reset was executed $1_B$ Kernel reset was executed
<b>0</b>	31:2	r	<b>Reserved</b> Read as 0; should be written with 0.

### **Kernel Reset Register 1**

The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

**Camera and ADC Interface (CIF)****BBB\_KRST1****Kernel Reset Register 1**(0018<sub>H</sub>)**Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															

r

rwh

Field	Bits	Type	Description
RST	0	rwh	<b>Kernel Reset</b> This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. The RST bit will be cleared (re-set to 0 <sub>B</sub> ) after the kernel reset was executed. 0 <sub>B</sub> No kernel reset was requested 1 <sub>B</sub> A kernel reset was requested
0	31:1	r	<b>Reserved</b> Read as 0 <sub>B</sub> ; should be written with 0 <sub>B</sub> .

**Kernel Reset Status Clear Register**

The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

**BBB\_KRSTCLR****Kernel Reset Status Clear Register**(001C<sub>H</sub>)**Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															

r

w

Field	Bits	Type	Description
CLR	0	w	<b>Kernel Reset Status Clear</b> Read always as 0 <sub>B</sub> . 0 <sub>B</sub> No action 1 <sub>B</sub> Clear Kernel Reset Status KRST0.RSTSTAT
0	31:1	r	<b>Reserved</b> Read as 0 <sub>B</sub> ; should be written with 0 <sub>B</sub> .

## Camera and ADC Interface (CIF)

### 26.3.14 CIF Programming Hints

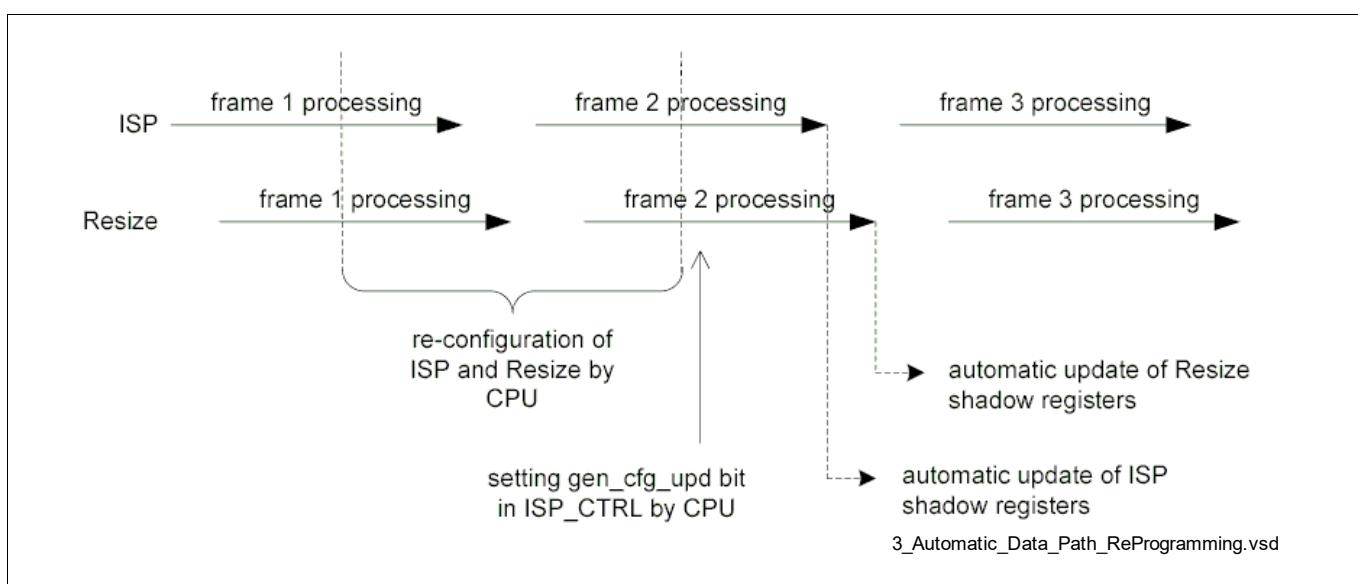
#### 26.3.14.1 Configuration and Shadow Registers

Most of the configuration registers inside the CIF hold static programming information (e.g. horizontal and vertical synchronization polarities at the sensor interface). But, in some cases it is required to dynamically re-program the active processing data path with new parameters (e.g. for zooming by re-defining the window). All of these new parameters must not be changed during the processing of a frame, but between frames or at the end of a frame.

Re-programming is done by a CPU attached to the Bus slave interface. Because it can not be ensured that the complete re-programming can be done during the vertical blanking period between two frames shadow registers have been introduced. These shadow registers hold the configuration values that are actually used for processing. Their corresponding configuration registers (e.g. ISP\_OUT\_H\_OFFSET\_SHD and ISP\_OUT\_H\_OFFSET) store the new values until the configuration update is triggered.

For initial programming after reset or for complete re-programming of the CIF (e.g. changing from viewfinder mode to snapshot mode and vice versa) there exists a cfg\_upd trigger bit in the main control register of each sub-module making use of shadow registers. Writing a 1 into this field copies the configuration register value into the shadow register. Mostly the main control register also has a shadow register. Other control bits set during the same write transfer as the cfg\_upd bit are also copied to the shadow register.

Another method of shadow register update is used for dynamically re-programming the data path (see [Figure 360 “Automatic Data Path Re-Programming” on Page 29](#)). In this case the ISP, which is always the first part of the pipeline, triggers the configuration update for the whole pipeline by setting the gen\_cfg\_upd bit in the ISP main control register ISP\_CTRL. This causes a configuration update pulse to be sent through the pipeline together with the last pixel of a frame. Each sub-module of the active processing pipeline updates its shadow registers after processing the last pixel of a frame and sends the configuration update pulse to the next stage of the pipeline.



**Figure 360 Automatic Data Path Re-Programming**

Some configuration registers exist that can be changed during processing without use of shadow registers. These are for example adjustment registers in the ISP or the color processing unit which have immediate influence of

## Camera and ADC Interface (CIF)

the processing in terms of pixel calculations. But, no processing errors or unknown formats will result in re-programming of these register. So it is safe to modify them at any time.

### 26.3.14.2 General Setup for Operation

The basic configuration of the CIF must be done in idle mode. This mode can be entered by either:

- Applying the asynchronous system reset
- Software reset which is combined with the asynchronous system reset
- Applying software resets to all CIF sub-modules (refer to VI\_IRCL register)
- Disabling the ISP by setting `isp_en` in `ISP_CTRL` to 0 and waiting for “`isp_off`” IRQ

In idle mode the following sub-modules must be configured: ISP, memory interface and control unit for image capturing to memory.

The configuration steps, which are performed by register accesses via the Bus slave interface, are the following:

- Set CIF to idle state
- Image acquisition configuration:
  - Set the input acquisition window depending on the attached camera device (`ISP_ACQ_H_OFFSET`, `ISP_ACQ_V_OFFSET`, `ISP_ACQ_H_SIZE`, `ISP_ACQ_V_SIZE`)
  - Define synchronization signal mode and polarity, data sample edge referring to the sensor clock (`ISP_ACQ_PROP`), and number of frames to be captured (`ISP_NR_FRAMES`)
- Image windowing setup:
  - Define the ISP output window (`ISP_OUT_H_OFFSET`, `ISP_OUT_V_OFFSET`, `ISP_OUT_H_SIZE`, `ISP_OUT_V_SIZE`)
- Immediate ISP configuration update (`cfg_upd = 1` in `ISP_CTRL`)
- Data path configuration:
  - Depending on the use case the data path must be configured for the desired operation (`VI_DPCL`)
- Main picture memory buffer configuration:
  - Define luminance (Y) buffer base address and size in system memory (`MI_MP_Y_BASE_ADDR`, `MI_MP_Y_SIZE`)
  - Define Y data start offset relative to the Y buffer (`MI_MP_Y_OFFSET`)
  - Define chrominance blue ( $C_B$ ) buffer base address and size in system memory (`MI_MP_CB_BASE_ADDR`, `MI_MP_CB_SIZE`)
  - Define  $C_B$  data start offset relative to the  $C_B$  buffer (`MI_MP_CB_OFFSET`)
  - Define chrominance red ( $C_R$ ) buffer base address and size in system memory (`MI_MP_CR_BASE_ADDR`, `MI_MP_CR_SIZE`)
  - Define  $C_R$  data start offset relative to the  $C_R$  buffer (`MI_MP_CR_OFFSET`)
- Immediate memory interface configuration update:
  - Define to update base and offset shadow registers(`init_base = 1` and `init_offset = 1` in `MI_INIT`)
  - Update shadow registers for video (in) and Bus part (out) (`cfg_upd_mi_in = 1` and `cfg_upd_mi_out = 1` in `MI_CTRL`)
- Enable/disable interrupts (`ISP_IMSC`, `MI_IMSC`)

Now the basic setup of all necessary registers is done.

For programming a specific use case the data path needs to be configured in the right way (please refer to chapter “[Use Case Description” on Page 47](#)). Additionally not active sub-modules can be switched off as described in chapter “[Power Management” on Page 51](#).

## Camera and ADC Interface (CIF)

### 26.3.14.3 Start-Stop Programming

#### 26.3.14.3.1 Data capturing controlled by the ISP

Data capturing is controlled by the ISP only. All the other sub-modules in the processing pipeline automatically stop processing if no video data is available.

The CIF starts and stops video data capturing in a frame synchronized way. So the time of enabling data capturing is not important as the ISP waits for the beginning of a frame before sampling data. The same applies for stopping video data capturing. Regardless of the current pixel position in the frame or outside the frame the current frame is always captured completely before disabling the ISP.

The ISP contains an input acquisition and an output formatting unit. Both units are able to start and stop video processing frame synchronously. For video capturing both units must be enabled, but for disabling capturing at least one of them must be switched off.

**Note:** *It is highly recommended to use the output formatter of the ISP for controlling video processing, because this unit is able to generate an “isp off” interrupt based on the ISP output frame after windowing.*

Additionally it is possible to specify the number of frames to be captured by writing this number into ISP\_NR\_FRAMES. Continuous capturing is enabled by writing a 0 into this register.

So to enable the ISP input\_en in ISP\_CTRL must be set to 1 first. At this point the ISP starts capturing data but does not provide them at its output interface. The output interface is part of the output formatter which also does the output windowing. It is enabled setting isp\_en in ISP\_CTRL to 1.

To stop video capturing it is sufficient to disable the ISP output formatter by setting isp\_en in ISP\_CTRL to 0, or if a certain number of frames to be captured have been specified, the ISP automatically disables its output formatter. Then the “isp off” interrupt is generated after putting out the last pixel of the last frame.

**Note:** *After stopping data capturing at the ISP the processing pipeline is still working for a while because of the pipeline delay of some pixel lines. This delay depends on the mode of operation. The maximum delay is about 11 lines.*

The pipeline is clean after either applying the software reset to the involved sub-modules (resulting in loss of data in memory), or it must be waited until the complete frame has been written into system memory. The latter is done by waiting for the “frame end” interrupt of the output unit.

### 26.3.14.4 Abort of Processing

Complete abort of processing can be either done by applying the asynchronous system reset to the CIF, the software reset for the entire CIF or by applying the software resets to all involved sub-modules which is the preferred method.

#### 26.3.14.4.1 Frame Skip

For the main picture data path another way of discarding pixel data exist: frame skipping. This can be used if e.g. the video encoding device can not keep up with the frame rate provided by the CIF. In this case the system controller has to set the skip bit in the MI\_INIT register of the memory interface.

For frame skipping the memory interface discards all remaining pixels of the current frame at its main picture input ports. It only completes the current burst transfer and flushes all main picture internal data FIFOs. Resetting of the related offset data pointers for the buffers depends on the value of init\_offset in MI\_CTRL. If init\_offset is

## Camera and ADC Interface (CIF)

set to 1 the buffer offset counters are reset to the programmed values in MI\_MP\_Y/CB/CR\_OFFSETS. If init\_offset is set to 0 the offset counters for all buffers are set to the beginning of the current (skipped) frame.

Frame skipping does not affect “frame end” interrupt generation.

### 26.3.14.4.2 Handling Picture Size Error

Aborting the ISP processing becomes necessary, if an input picture size violation is detected by the hardware. In this case the interrupt PIC\_SIZE\_ERR is raised, and the ISP does not generate a frame end interrupt, even if the picture size is corrected. To recover from this condition, a software reset of the ISP is necessary. After a reset the picture size needs to be programmed to the appropriate values and processing can be started again.

To find the reason behind the picture size error the register ISP\_ERR might be helpful. This register provides the information about where exactly the size error occurred.

### 26.3.14.5 Interrupt Handling

The CIF provides two programmable level active interrupt outputs driven by the ISP and memory interface sub-modules. Interrupt control and status registers are located locally in their originating sub-modules. All sub-module masked interrupts are ORed together to drive the interrupt line.

Five interrupt registers are provided for each interrupt line per block able to generate interrupts:

- Interrupt mask register (IMSC)
- Raw interrupt status register (RIS)
- Masked interrupt status register (MIS)
- Interrupt clear register (ICR)
- Interrupt set register (ISR)

The IMSC register defines if the events actually generate an interrupt. If a mask bit is set, the corresponding event is allowed to generate an interrupt. Per default all mask bits are set to 0 to prevent interrupt generation.

The RIS register shows the status of all sub-module internal events without effect of the mask register IMSC.

The MIS register shows the status of the events allowed to trigger an interrupt (MIS = RIS AND IMSC). If an interrupt has been thrown the MIS register must be read for locating the accountable event.

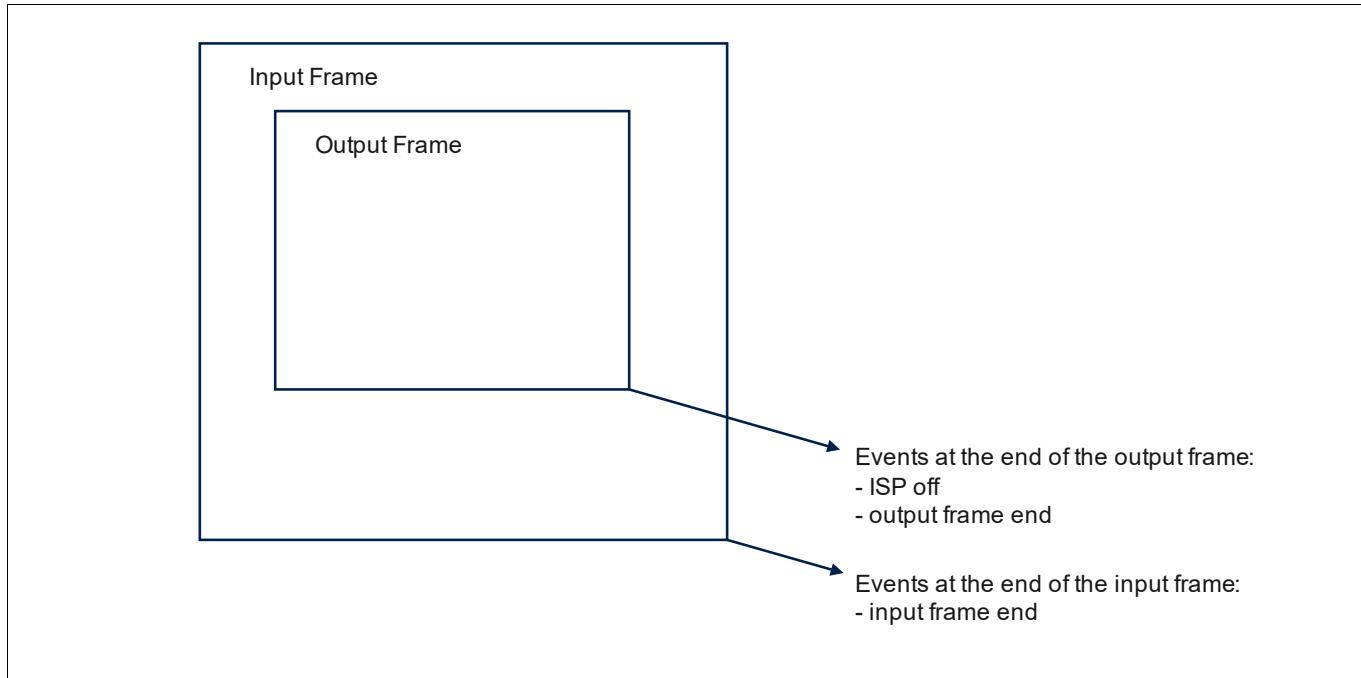
The ICR register is used for clearing interrupts by resetting the event. Writing a 1 into a interrupt bit resets the respective event. Reading of this register always returns “ $00_H$ ”.

The ISR register is used to trigger events for debug and test purposes. Writing a 1 into an interrupt bit activates the associated event and an interrupt will be thrown if the mask register is set accordingly. Reading this register always returns “ $00_H$ ”.

### 26.3.14.5.1 ISP Events

The ISP generates events at several positions during input and output data formatting. [Figure 361](#) shows the trigger points for the following events:

- ISP returns to idle state (isp\_off)
- Output frame end reached (out\_frame\_end)
- Input frame end reached (in\_frame\_end)

**Camera and ADC Interface (CIF)****Figure 361 Frame Based ISP Events**

Additionally the following error events may be generated:

- Input picture size violation detected (pic\_err)
- Loss of data detected during an active (data\_loss)

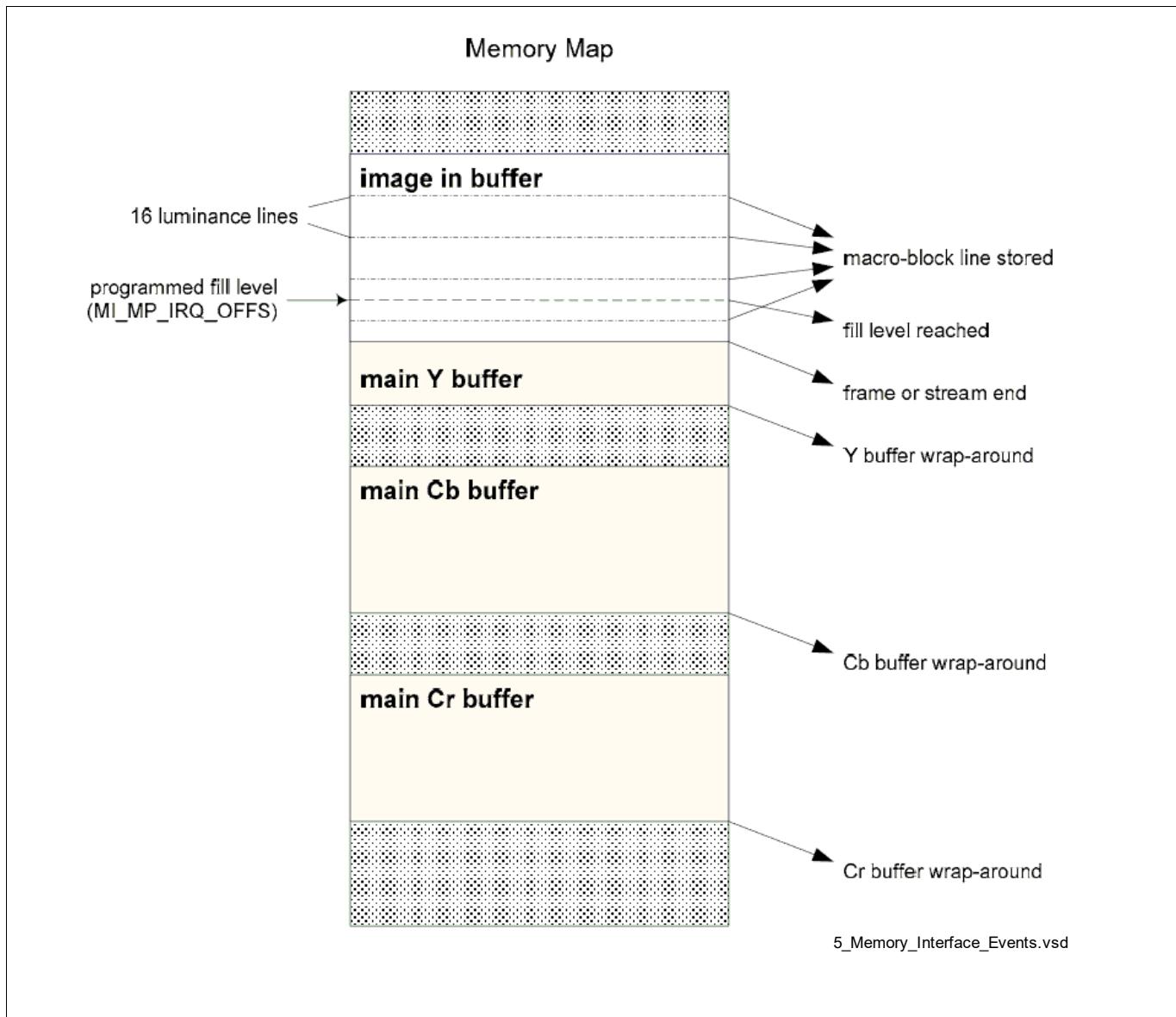
These two events are immediately triggered as they occur. For more information on ISP events please refer to chapter "[“ISP Programming” on Page 35](#).

## Camera and ADC Interface (CIF)

### 26.3.14.5.2 Memory Interface (MI) Events

The MI provides events from several internal sources (see [Figure 362 “Memory Interface Events” on Page 34](#)) supporting video encoding as well as signalling error and transfer end conditions:

- Bus write error detected (write\_err)
- Main picture buffer wrap-around occurred for Y, C<sub>b</sub>, C<sub>r</sub> buffers (wrp\_m\_y, wrp\_m\_cb, wrp\_m\_cr)
- Programmed fill level reached in main Y buffer (fill\_m\_y)
- Next macro-block line stored (mblk\_line)
- Last pixel of image or RAW data stream is written into memory (frame\_end)



**Figure 362 Memory Interface Events**

The “macro block line” and “frame end” events not only depend on the Y buffer, but also on C<sub>b</sub> and C<sub>r</sub> buffers because all pixel components have to be written to memory before triggering the appropriate event. For “frame end” this synchronization works for all supported YC<sub>b</sub>C<sub>r</sub> formats (4:2:2, 4:2:0). The “macro-block line” event relies on 16 luminance and 8 chrominance lines, which are in YC<sub>b</sub>C<sub>r</sub> 4:2:0 format.

## Camera and ADC Interface (CIF)

For better debugging in MI an additional status register is implemented. The MI shows status information of the FIFOs in the MI\_STATUS register and if a read or write Bus error occurred. This registers can be cleared with the according MI\_STATUS\_CLR register. Usage of this register should be as follows: If an error interrupt was triggered from the sensor FIFO of the ISP (DATA\_LOSS interrupt), then the MI\_STATUS register should be checked to find out the reason for the error. One reason could be the latency of the Bus bus, which results in FIFO full bits set in the MI\_STATUS register.

### 26.3.14.6 Reset Handling

Three types of resets are supported by the CIF:

- Asynchronous system reset
- Software reset for the entire CIF
- Independent software resets for each sub-module

The asynchronous reset is a low active system reset for resetting all flip-flops inside the CIF. After an asynchronous reset no processing, status, or configuration information is kept.

The same behaviour as for “asynchronous system reset” is true for the software reset bit of the entire CIF. All status and configuration register information is reset. After applying this reset the software must wait at least 10 module clock cycles before the CIF can be reprogrammed.

The independent software resets for each sub-module are meant for just resetting processing and status information, but not configuration register content. So, the software reset is first choice if a processing error occurs and the system should be re-initialized without need to fully re-program the device. It is triggered by single bits for all CIF sub-modules in VI\_IRCL.

### 26.3.14.7 Programming Guide

This chapter gives a detailed overview about programming of the particular CIF sub-modules.

#### 26.3.14.7.1 ISP Programming

The ISP sub-module contains input acquisition, output formatting, as well as status and error interrupt generation.

An overview of the internal processing blocks of the ISP is given in the figure [“Block Diagram of the ISP Sub Module” on Page 8](#).

#### Error Handling

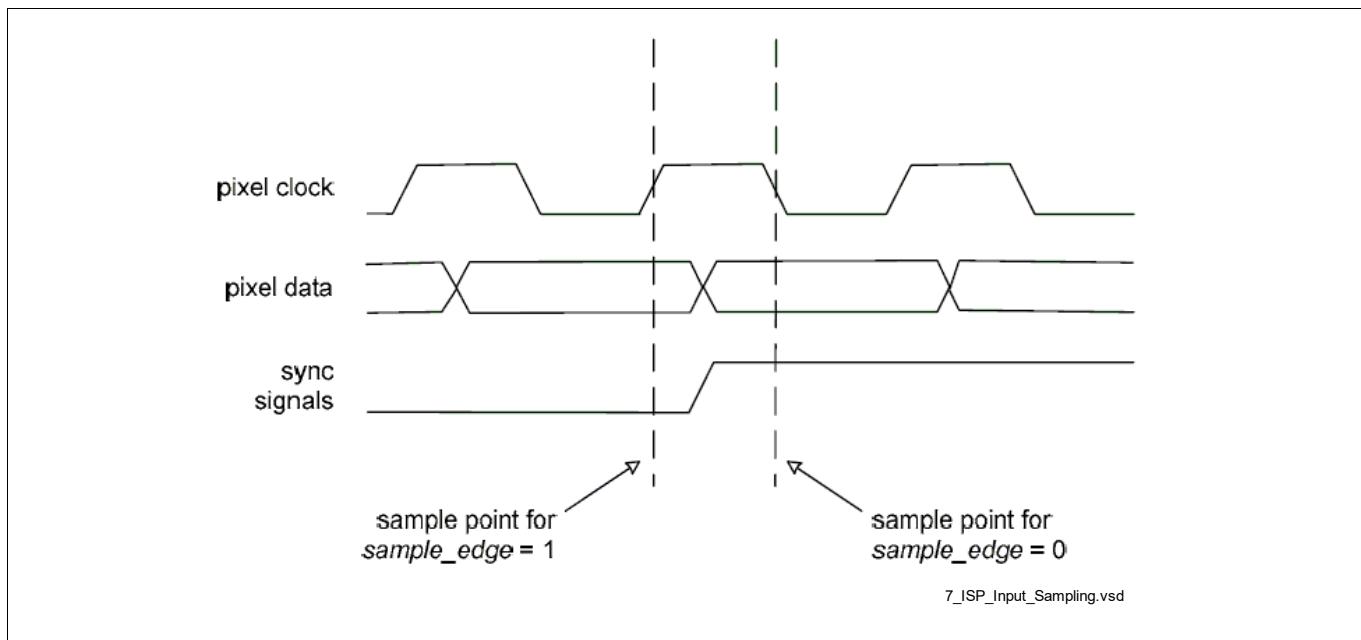
If an input picture size violation was detected and reported (interrupt) by the ISP, then the ISP needs to be reseted (soft-reset) in order to do the processing of the following frames. Without reset the frame end interrupt will not work correctly for the following frames, even if the picture size is correct then.

#### Input Acquisition

The input formatter is responsible for sampling data from the sensor device and providing it to the other blocks in the processing pipelines.

As the clock provided by the image sensor is used for data sampling, the sample edge can be selected depending on the phase shift between data and pixel clock. If sample\_edge in ISP\_ACQ\_PROP is set to 1 the rising edge of the pixel clock is used, otherwise the falling edge. The sample edge selection depends on the phase relationship between pixel clock and image data.

## Camera and ADC Interface (CIF)

**Figure 363 ISP Input Sampling**

The input acquisition has to be adapted to the specifics of the attached sensor device in terms of image resolution, color space format, black pixel regions, etc.

In addition please make sure that the sensor device delivers at least 2 lines of vertical blanking and at least 5 pixels horizontal blanking in its video timing. This is needed for the CIF ISP to function properly.

The input acquisition window has to be defined using the registers

ISP\_ACQ\_H\_OFFSET,  
ISP\_ACQ\_V\_OFFSET,  
ISP\_ACQ\_H\_SIZE,  
ISP\_ACQ\_V\_SIZE.

In addition the sample window can be further cropped for the following processing chains by the registers

ISP\_OUT\_H\_OFFSET,  
ISP\_OUT\_V\_OFFSET,  
ISP\_OUT\_H\_SIZE,  
ISP\_OUT\_V\_SIZE.

This is necessary, because processing blocks must not be fed with images containing black pixel areas. These areas have to be cut off before the image data is transferred to the CIF's ISP.

Furthermore the polarity of the horizontal and vertical synchronization signals in ITU-R BT.601 mode have to be defined using hsync\_pol and vsync\_pol in ISP\_ACQ\_PROP.

Limited support for capturing of interlaced video is available. So either odd or even, or both fields can be sampled (field\_sel in ISP\_ACQ\_PROP).

In the ISP\_CTRL programming register the operating mode for the ISP must be selected with isp\_mode (also refer to the following chapters).

Input formatter enabling and disabling is described in chapter [“Start-Stop Programming” on Page 31](#).

### Output Formatting and Image Stabilization

At the output of the ISP sub module the image data may be cropped a third time by using the registers

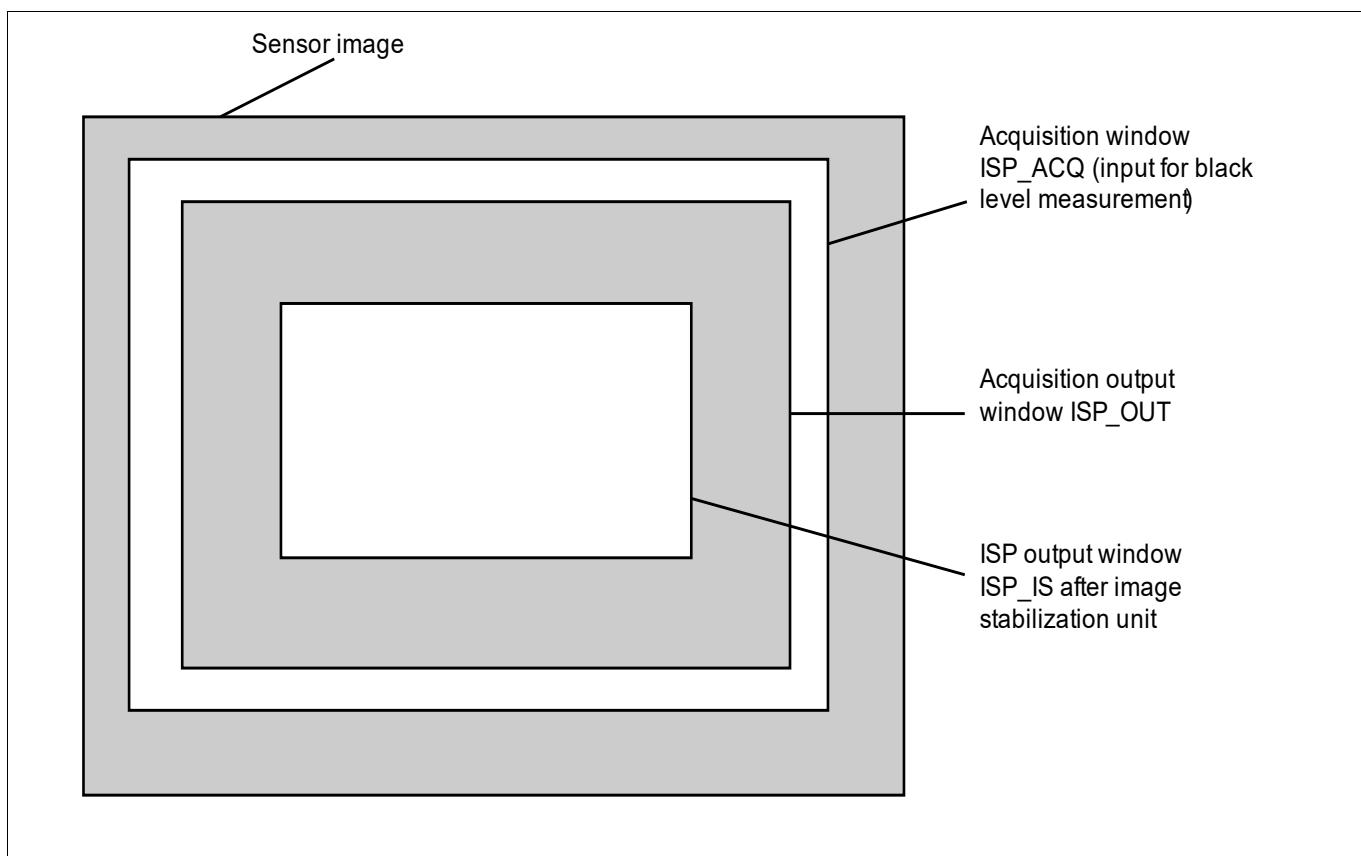
ISP\_IS\_H\_OFFSET,

## Camera and ADC Interface (CIF)

ISP\_IS\_V\_OFFSET,  
ISP\_IS\_H\_SIZE,  
ISP\_IS\_V\_SIZE.

These registers should be used, if digital zoom features shall be implemented in the system software.

**Figure 364** illustrates the different possible regions which can be defined by the corresponding register settings. Some of these regions may have the same size and offset (settings), if a differentiation is not needed.



**Figure 364 Possible cropping regions of ISP**

In addition the output cropping module is designed to support image stabilization features for video capturing. For image stabilization a sub frame must be chosen from the input frame, because a certain border around the output image is needed to have some margin for correcting the position of the image. Based on externally generated camera global motion data, image stabilization offers the functionality to compensate for that camera motion by moving the chosen sub frame across the input frame according to the signalled camera motion. The information source for global motion may be a motion sensor or extracted from the image content by a video encoder. The global motion vector has to be written into the configuration register ISP\_IS\_DISPLACE by software for each frame. To prevent the sub frame from running out of the input frame, the sub frame is re-centered at a programmable rate that is proportional to its distance from the center of the input frame. The rate is programmed by setting the register ISP\_IS\_RECENTER with appropriate values.

In addition, the maximum allowed displacement of the sub window is programmable by using ISP\_IS\_MAX\_DX and ISP\_IS\_MAX\_DY registers. The output picture size is using shadow registers. If the programmed size can't be reached due to incorrect register programming or wrong input formats, a picture size error interrupt is triggered.

In standard mode (image stabilization switched off: is\_en=0 in ISP\_IS\_CTRL), the registers ISP\_IS\_H\_OFFSET, ISP\_IS\_V\_OFFSET, ISP\_IS\_H\_SIZE, and ISP\_IS\_V\_SIZE are used to cut a programmable sub frame out of the input frame as described above.

## Camera and ADC Interface (CIF)

If image stabilization is switched on, the horizontal and vertical offset of the chosen sub frame is automatically updated. This automatic update according to camera displacements dx and dy programmed to the ISP\_IS\_DISPLACE register is implemented as described below.

Instead of using the values programmed to the configuration registers ISP\_IS\_H\_OFFSETS and ISP\_IS\_V\_OFFSETS, (which are now used to configure the initial position of the chosen sub frame) two new internal registers cur\_h\_offs and cur\_v\_offs are used to dynamically choose the sub frame sent to the output interface. These are updated with each frame in the following way:

```

if IS_RECENTER != 0
    cur_h_offs = (cur_h_offs - dx) - ((cur_h_offs - IS_H_OFFSETS) / 2IS_RECENTER)
else
    cur_h_offs = (cur_h_offs - dx)

cur_h_offs is clipped to the range
[IS_H_OFFSETS - IS_MAX_DX ... IS_H_OFFSETS + IS_MAX_DX].

```

The system software has to ensure that IS\_H\_OFFSETS is greater or equal IS\_MAX\_DX and that IS\_H\_SIZE + IS\_H\_OFFSETS + IS\_MAX\_DX does not exceed the horizontal size of the input frame. Recommended value for IS\_MAX\_DX is equal to IS\_H\_OFFSETS.

```

if IS_RECENTER != 0
    cur_v_offs = (cur_v_offs - dy) - ((cur_v_offs - IS_V_OFFSETS) / 2IS_RECENTER)
else
    cur_v_offs = (cur_v_offs - dy)

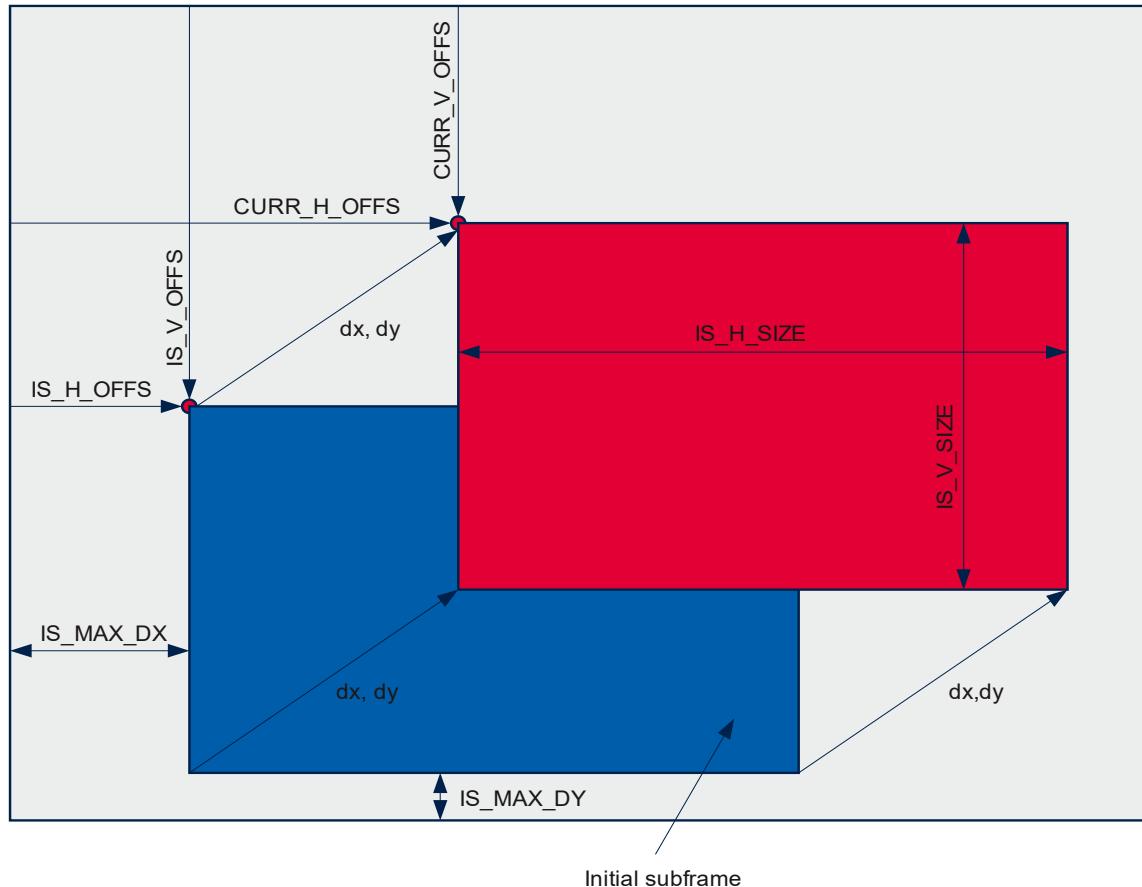
cur_v_offs is clipped to the range
[IS_V_OFFSETS - IS_MAX_DY ... IS_V_OFFSETS + IS_MAX_DY].

```

The system software has to ensure that IS\_V\_OFFSETS is greater or equal IS\_MAX\_DY and that IS\_V\_SIZE + IS\_V\_OFFSETS + IS\_MAX\_DY does not exceed the vertical size of the input frame. Recommended value for IS\_MAX\_DY is equal to IS\_V\_OFFSETS.

The system software has to ensure that the dx and dy values in ISP\_IS\_DISPLACE register are always up to date. Especially when no movement was detected the values have to be set to zero.

## Camera and ADC Interface (CIF)



**Figure 365 Illustration of image stabilization parameters**

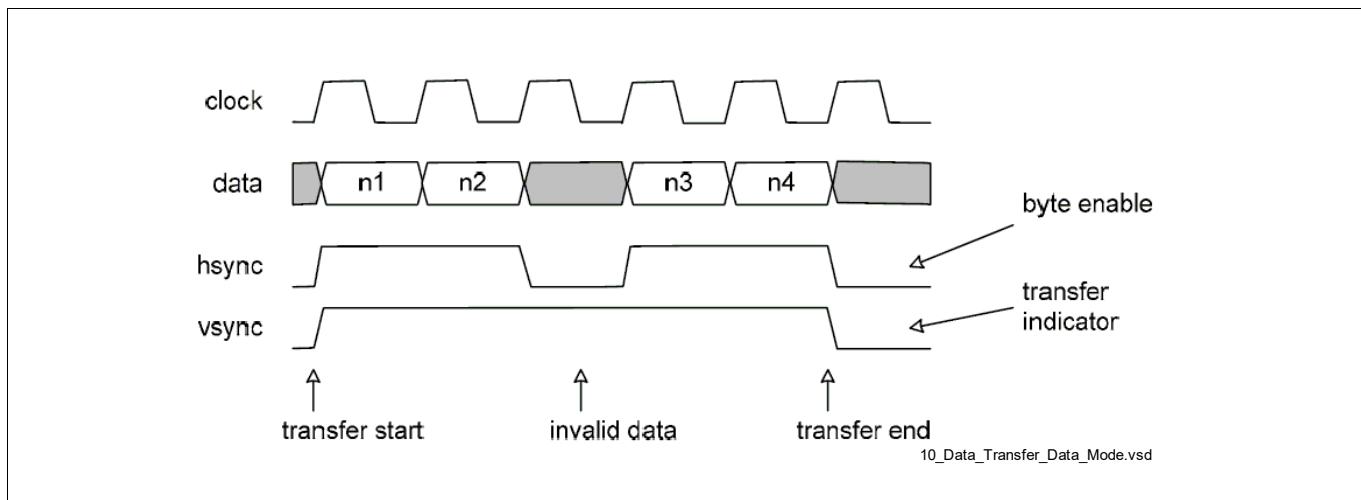
### Data Mode

In data mode all input data are considered as bytes. No line or frame organization is expected any more, so the states of the hsync and vsync hardware lines are considered as "byte enable" and "transfer indicator". This mode allows connecting the CIF to any source delivering a data stream (e.g. camera sensor including a JPEG encoder). This mode is selected by programming a "4" into the `isp_mode` field of `ISP_CTRL`. In this mode no limitation for the amount of data exists. The data will be stored in system memory in the region "main Y buffer".

As the synchronization signal polarities are programmable, the same applies to byte enable and transfer indicator as well. Also the sample edge can be selected by programming the register `ISP_ACQ_PROP` accordingly.

To start a transfer sequence, the transfer indicator (vsync) and byte enable (hsync) may be asserted (driven to an active level, usually low) at the same time or one after another in the following order: first assert vsync, then assert hsync. The transfer end condition is detected by the CIF when these signals are de-asserted as shown in **Figure 366**. Detecting the transfer end condition will lead to a main-path frame-end interrupt as soon as the last byte of this transfer is written to the system memory. During a transfer sequence, sampling of the input data can be paused by de-asserting the hsync for the desired amount of clock cycles.

## Camera and ADC Interface (CIF)



**Figure 366 Data Transfer in Data Mode with Active High Level**

Note that because there are no line and frame structures in the data mode any more, there is also no frame synchronized start mechanism as described above. Thus, the time of enabling the ISP is important here, because if the transfer indicator line is asserted at the moment of enabling the ISP, data capturing would start right away. If the entire amount of bytes of a transfer sequence is to be captured, the ISP must be enabled while the transfer indicator is de-asserted.

In situations with periodically repeated transfer sequences, the suggested strategy of capturing one entire sequence is as follows:

- Setup the complete processing chain including input acquisition, data path and output buffer size and address.
- Setup the memory interface to skip the next incoming frame (i.e. suppress the actual writing in system memory)
- Start the ISP for one frame only and wait for the frame-end interrupt. Because starting the ISP was done unsynchronized with the transfer sequence repetition, the CIF usually will not sample the entire number of bytes of this sequence. But this is not important, because we have suppressed writing to memory anyway, and are only interested in the frame-end-notification interrupt.
- As soon as the frame-end interrupt occurs, we know for sure that the last transfer sequence has just been completed. Now re-program the output buffer address, start the ISP for another single frame, and wait again for the frame-end notification.
- Upon occurrence of the frame-end interrupt, the entire number of data bytes from the last transfer sequence can be found in the output buffer.

In data mode, the values of the registers ISP\_ACQ\_V\_OFF, ISP\_ACQ\_H\_OFF, ISP\_ACQ\_H\_SIZE and ISP\_ACQ\_V\_SIZE are irrelevant.

### RAW Picture Mode

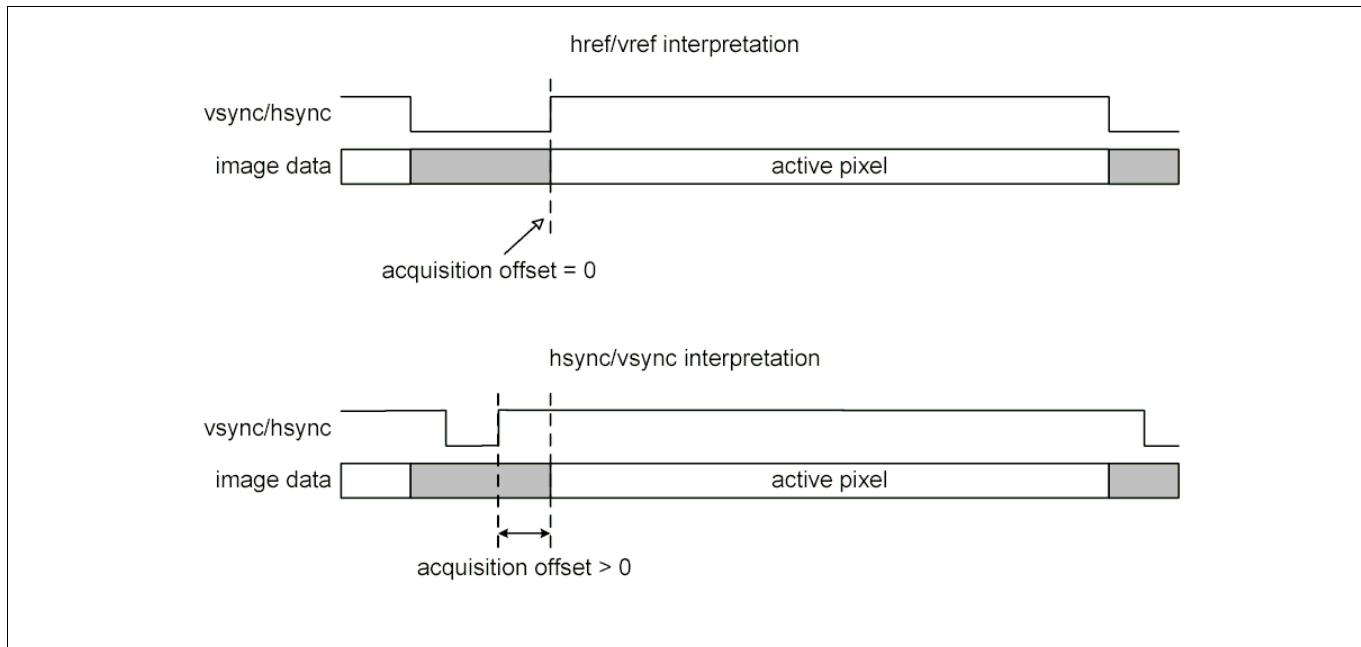
RAW picture mode allows sampling of the input image according to the setting of the BT.601 sync signals and providing it directly at the output of the ISP. No data processing is done at all allowing to store the 8 up to 16-bit data samples from the sensor device directly in system memory. No luminance/chrominance separation and re-ordering are done. After writing a 0 into the isp\_mode field of ISP\_CTRL the ISP is set to RAW picture mode.

The maximum size of data is limited by the offset and size counters. This means a maximum of 4095x4095 bytes are possible. The image data will be stored in the main Y buffer. In case of more than 8 bits data, the data bits are stored in 16-bit words in memory. The bits are stored MSB aligned.

### ITU-R BT 601/656 YC<sub>b</sub>C<sub>r</sub>4:2:2 Mode

## Camera and ADC Interface (CIF)

This ITU-R BT 601/656 mode allows the connection of an image sensor with integrated ISP functionality. In this case  $YC_bC_r$  data is expected at the sensor interface. In ITU-R BT 656 mode the synchronization signals are not considered as all timing reference data are encoded into the data stream (SAV and EAV codes). For ITU-R BT 601 mode the vertical and horizontal synchronization signals are sampled together with the image data. The synchronization signals can be interpreted as synchronization pulses, or as reference signals to be seen in the following figure.



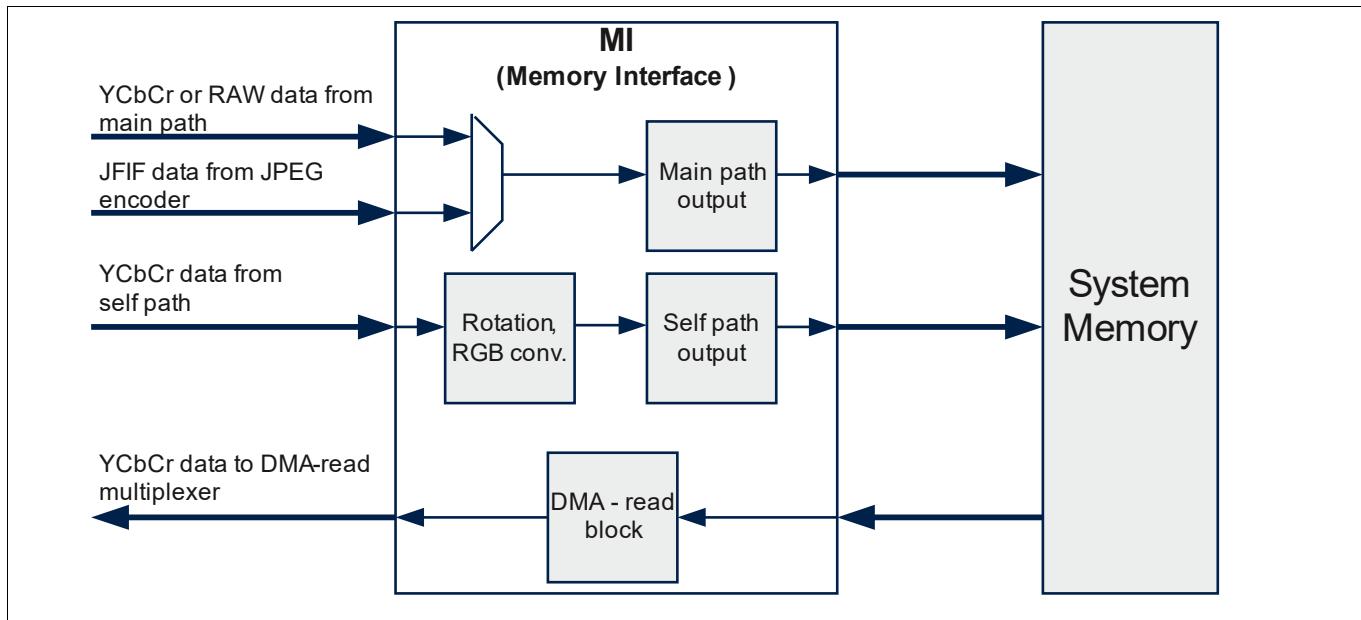
**Figure 367 Reference and Synchronization Signals for ITU-R BT 601 Mode**

### 26.3.14.7.2 Memory Interface Programming

The CIF memory interface unit (MI) is responsible for reading/writing image data from/to the system memory. As shown in Figure “[Memory Interface Unit overview](#)” on Page 42, the MI has three main tasks, which can operate independently:

- Take image data from the main path, either in YCbCr or RAW format and write it into certain data buffers located in the system memory.

## Camera and ADC Interface (CIF)



**Figure 368 Memory Interface Unit overview**

### External Output Buffers

Depending on which paths are enabled and which output format is selected, the MI unit can simultaneously output image data to up to 6 independent buffers. The buffers are named according to their relation to the main data path and their usage in the (most common) YCbCr planar output format: main-Y-buffer, main-Cb-buffer, main-Cr-buffer.

The buffers itself are not part of the CIF IP; they are system memory areas accessible from the bus that is also connected to the CIF. One can think of the MI as a kind of DMA controller, which writes to the configured memory locations without any CPU assistance. To configure the memory locations, each of the buffers mentioned earlier has a set of registers assigned to it.

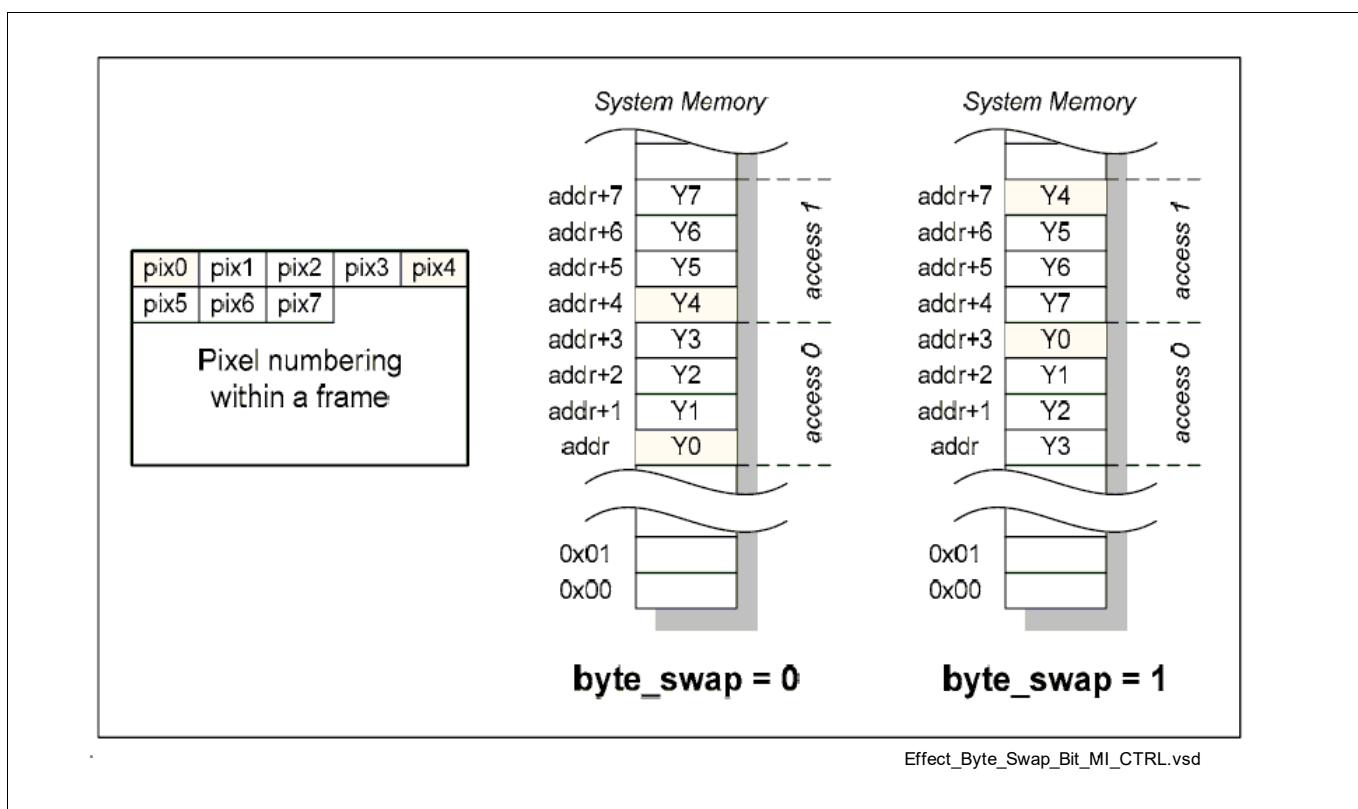
- MI\_xx\_xx\_BASE\_AD\_INIT
  - This is the base address of the memory area to use for that buffer. (e.g. the lowest address of that buffer)  
Note that buffer base addresses must be 4-byte-aligned. In case the target system uses some kind of PMMU, the physical address must be specified, because virtual-to-physical translation is not possible for this kind of DMA-like transfer.
- MI\_xx\_xx\_SIZE\_INIT
  - This is the size of the memory area in bytes. Note that the size is also required to be a multiple of 4. It is ensured that for write operations in this particular buffer, the MI will only write to memory locations from the base address to the base address + the size - 1. If the MI reaches the end of the memory area while continuously writing data, the buffer "wraps around" and it proceeds to write starting from the base address again. This wrap around event can also be used to trigger an interrupt. See the MI\_RIS register for details.
- MI\_xx\_xx\_OFFSET\_CNT\_INIT
  - This is the initial offset (in bytes) for writing to the buffer after e.g. a soft reset. Usually, to write to the buffer from the very beginning, an offset of zero is to be programmed into this register.
- MI\_xx\_xx\_OFFSET\_CNT\_START
  - This is a read-only register, which holds the offset at which the MI had written the last processed frame. This register is updated at frame end, so at any given time, it contains always the starting offset of the last completely processed frame.

## Camera and ADC Interface (CIF)

One of the buffers, the main-Y-buffer, is additionally able to trigger an interrupt as soon as a particular filling level is reached. This level can be configured in terms of an offset count (in bytes) from the base address using the MI\_MP\_Y\_IRQ\_OFFSET\_INIT register.

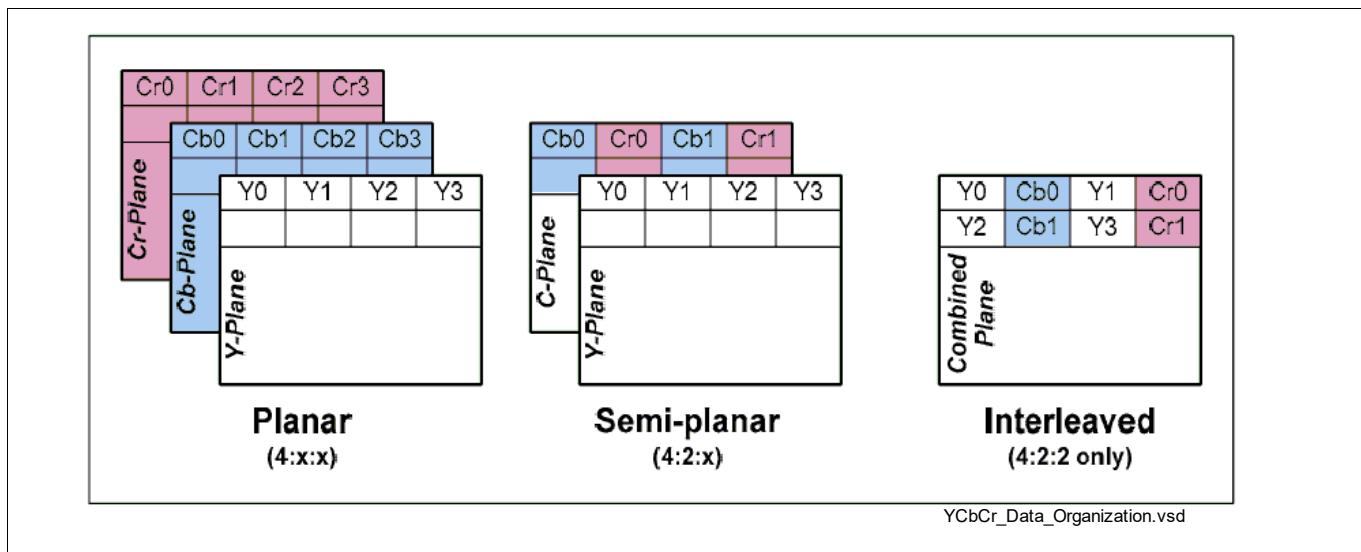
Further, the MI supports updating the buffer configuration seamlessly between two consecutive frames without dropping a frame by using the shadow register concept described in chapter [“Configuration and Shadow Registers” on Page 29](#). Two bits in the MI\_CTRL register (init\_offset\_en and init\_base\_en) can be used to select whether the base address and size, or the offset counters, or both are being updated at the next configuration update pulse. It should be noted that it is only possible to update the selected parts of all the buffers at once. It is not possible to re-configure only specific buffers and leave the others left untouched.

As mentioned earlier, there are some restrictions permitting only 4-byte-alignment buffer base addresses, and both sizes and offsets also need to be a multiple of 4. This is because the MI writes the data to the memory areas in 4-byte-accesses. It is possible to change the endianess of these accesses with the byte\_swap bit in the MI\_CTRL register. [Figure 369](#) illustrates this with the example use case of writing 8-bit Y-samples from a frame with a width of 5 pixels. The setting of the byte\_swap bit affects the write accesses to all buffers regardless of whether YCbCr or RAW data is to be written. Note that "addr" is a 4-byte aligned address in this example. For all other output schemes illustrated and discussed in this chapter, a setting of byte\_swap = 0 is assumed.



**Figure 369 Effect of the byte\_swap bit in MI\_CTRL**

## Camera and ADC Interface (CIF)



**Figure 370** YCbCr Data organization

### Main Path output programming

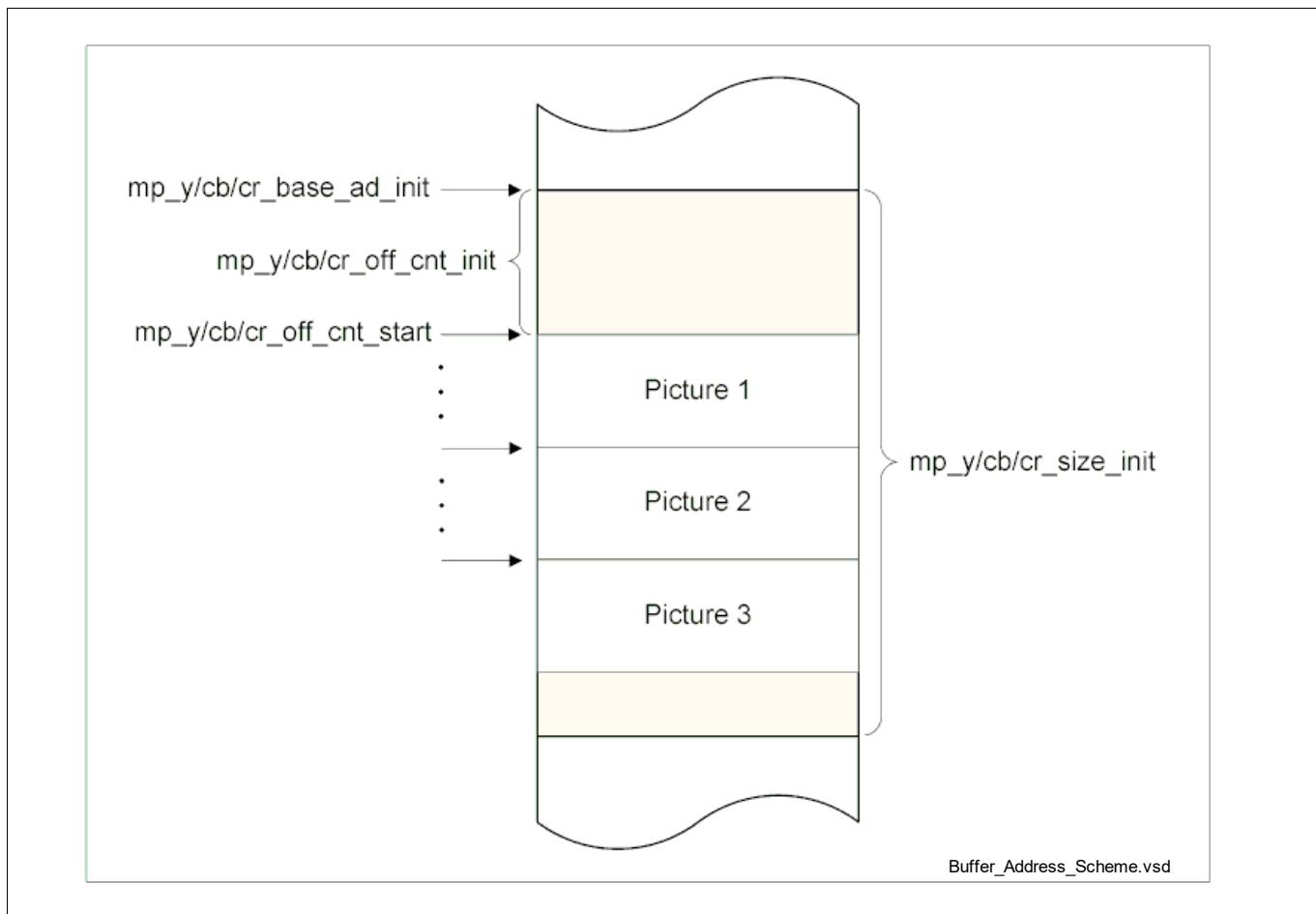
The main path output sub-module of the MI is intended to be used for video stream generation (YCbCr for further compressing via software encoders), or high resolution still image snapshot. To achieve this, it can be configured to output one out of the following data sources:

- YCbCr data from the main path scaler
  - To enable this, bit mp\_enable in the MI\_CTRL register is to be set. The data can be output in planar, semi-planar or interleaved format, as shown in [Figure 370](#). The selection is done with the bits mp\_write\_format in the MI\_CTRL register. In semi-planar mode, the main-Cb-buffer is used for the multiplexed Cb/Cr values and the main-Cr-buffer remains unused. In interleaved mode, only the main-Y-buffer is used for all the multiplexed values.
- RAW data from the main path scaler.
  - This mode is enabled by setting the bit raw\_enable in the MI\_CTRL register. RAW data is assumed to be received via the same way as YCbCr data, but is written to the main-Y-buffer only. In RAW mode, the mp\_write\_format bits of the MI\_CTRL register are used to select between 8-bit mode (which will write only the 8 most significant bits of every sample) up to 16-bit mode (which will e.g. write the full 16-bit msb-aligned into 2 bytes of the output buffer). This implies that for the higher than 8-bit modes, a buffer size twice as much is necessary as for the 8-bit mode, and the least significant bits of every 16-bit value are unused and remain zero.

In some RAW data modes the actual number of bytes to write per frame depends on the input signal and/or the compression ratio, and is not known in advance. Thus, this number is given in the read-only MI\_BYTE\_CNT register, which is updated at frame end only. So this register always shows the number of bytes written to the main-Y-buffer for last completely processed frame.

It is assumed that all the sub-modules within the CIF's processing chain located in front of the MI are configured in a way that the input data to the MI main path corresponds to what it is configured for. If this is not the case, the behavior of the MI is not predictable.

## **Camera and ADC Interface (CIF)**



**Figure 371 Buffer address scheme, valid for main picture buffers**

### **26.3.14.7.3 Getting Started - First steps for startup**

This chapter describes the first steps to startup the CIF in YCbCr mode. The procedure and the register contents are presented. Generally, all sub modules (like MI, EP, LDS) within CIF can be configured in arbitrary order but the ISP sub module must be enabled last.

## General Hints

Activity of the input port (data and sync lines) can be observed through register ISP\_FLAGS\_SHD.

## Recipe 1: YCbCr Bypass Mode

In YCbCr Bypass mode the image sensor is expected to deliver pre-processed image data in YCbCr format at its parallel output port. Because of the pre-processing, most of the CIF's ISP functionality can be bypassed, so only a small number of registers must be programmed to put this mode into operation. The Main Scaler unit will also be bypassed, so that the incoming data will be presented almost unmodified at the output. All that's left to do in the processing chain is YC-sequence reordering.

## Requirements / Preconditions

In this example, we assume a sensor delivering VGA-sized (640x480) frames continuously using YCbCr data format with 4:2:2 color sub sampling. Synchronization is assumed to be done according to ITU.BT601 (i.e. using separate wires carrying hsync and vsync signals, both active high in this example).

## Camera and ADC Interface (CIF)

It is further assumed that the sensor is already up and running and that it is configured to the mode described above. Its pixel clock is running and meets the CIF's AC-requirements, and the sync signals and image sensor data bus are routed to the CIF's input ports.

### Basic Initialization

1. Enable the main clock in VI\_CCL  $VI\_CCL = 0000\ 0000_H$
2. Enable all clocks of all sub-modules  $VI\_ICCL = FFFF\ FFFF_H$
3. Trigger the asynchronous system reset of the CIF

Typically the asynchronous reset line is connected to a central module of your SOC, handling resets for all components. Usually there is one bit reserved for this purpose.

There may be some delay required in the software before the following access to the CIF (time until the central module clears reset, time until the reset signal is synchronized to the sensor clock domain of the CIF).

### ISP input acquisition

1. Switch to ITU.BT601 compatible mode (ISP still disabled yet)  $ISP\_CTRL = 0000\ 0004_H$
2. Set the input acquisition properties according to what is delivered by the sensor: sampling at falling edge of pixel clock; h- and v-sync high active; sample sequence Y-Cb-Y-Cr; don't care about odd or even fields; 8-bit data interface.  
 $ISP\_ACQ\_PROP = 0000\ 0300_H$
3. Set the input acquisition window according to what is delivered by the sensor.(here, we assume no delay between the sync signals and the active pixel data, that's why the offset values can remain zero. Horizontal size is twice the image width, because for every 2 incoming pixels, there are also 2 chrominance values):

$ISP\_ACQ\_H\_SIZE = 0000\ 0500_H$  ( $2 * 640 = 1280$ )

$ISP\_ACQ\_H\_OFS = 0$

$ISP\_ACQ\_V\_SIZE = 0000\ 01E0_H$  (480)

$ISP\_ACQ\_V\_OFS = 0$

### ISP output formatter

1. We want to output the complete incoming frame from the sensor (no cropping) so the whole sensor resolution need to be programmed in the output window

$ISP\_OUT\_H\_SIZE = 0000\ 0280_H$  (640)

$ISP\_OUT\_V\_SIZE = 0000\ 01E0_H$  (480)

$ISP\_OUT\_H\_OFFS = 0$   $ISP\_OUT\_V\_OFFS = 0$

### Data Path

1. Select parallel interface and main data path, enable output of main data path

$VI\_DPCL = 0000\ 0001_H$   $MI\_CTRL = 0000\ 0001_H$

### Memory Interface

1. Set Y, Cb and Cr buffer position and size (32bit aligned). Update shadow registers with next configuration update:
  - a)  $MI\_MP\_Y\_BASE\_AD\_INIT = <Y\ base\ address>$
  - b)  $MI\_MP\_Y\_SIZE\_INIT = <sizeY>$
  - c)  $MI\_MP\_CB\_BASE\_AD\_INIT = <Cb\ base\ address>$
  - d)  $MI\_MP\_CB\_SIZE\_INIT = <sizeC>$
  - e)  $MI\_MP\_CR\_BASE\_AD\_INIT = <Cr\ base\ address>$

## Camera and ADC Interface (CIF)

f) MI\_MP\_CR\_SIZE\_INIT = <sizeC>

Set to main picture data mode (do not swap bytes, do not mirror) and update shadow register immediately. (This also transfers the base address and the size to their respective shadow registers).

MI\_CTRL = 0030 0001<sub>H</sub>      MI\_INIT = 0000 0010<sub>H</sub>

### Interrupt Processing

1. Setup interrupt service routines, clear and enable interrupts if needed, e.g. frame end interrupt of memory interface:

a) MI\_ICR = 0000 0001<sub>H</sub> MI\_IMSC = 0000 0001<sub>H</sub>

### Start the CIF

1. Enable input acquisition and output formatter (to start the output formatter there is no configuration update necessary). As soon as the CIF is started the lower two bits of ISP\_FLAGS\_SHD are set.

ISP\_CTRL = ISP\_CTRL | 00000011<sub>H</sub> With the initialization sequence above, the CIF is set to sample all incoming frames and presents them at the output. If this is not the desired use case and e.g. a single frame is to be captured instead, this can be configured using the input acquisition register ISP\_ACQ\_NR\_FRAM. If this register is nonzero, only the given number of frames is being captured after the CIF has been started.

### 26.3.14.8 Use Case Description

The CIF is designed to support for example the following mobile phone use cases:

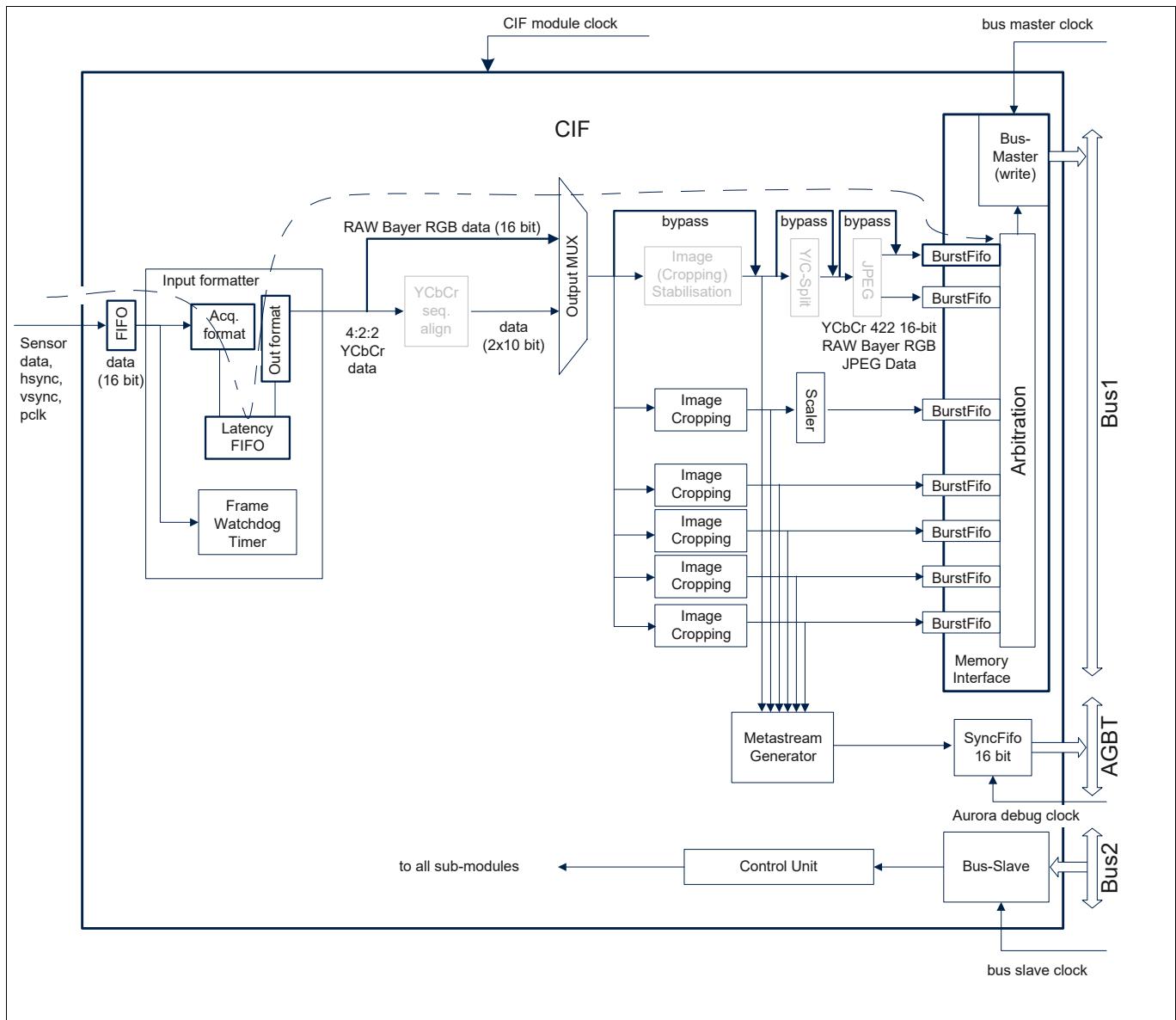
- Data transfer from external devices into system memory
- Viewfinder mode for still image capture
- Still image capture
- Video encoding

The above mentioned use cases are described in detail in the following chapters. Furthermore information about power management and configuration accesses are also given in this section.

## Camera and ADC Interface (CIF)

### 26.3.14.8.1 Data Transfer

This use case transfers byte data, like JPEG compressed data from a camera sensor with integrated JPEG encoder, into system memory (main Y buffer). Further processing must not be done as these data are no plain image or video data. The [Figure 372](#) shows the blocks active during data transfer. The inactive blocks are bypassed or disabled.



**Figure 372 Active Blocks for Use Case Data Transfer**

A transfer indicator and a byte valid signal (refer to [Figure 366](#)) are used to identify transfer start and stop as well as to identify valid data bytes. The ISP has to be set to data mode (mode field in ISP\_CTRL), all other functions (e.g. offset and size registers) are not used. The data path must be switched to 8 bit data/Raw data mode (VI\_DPCL).

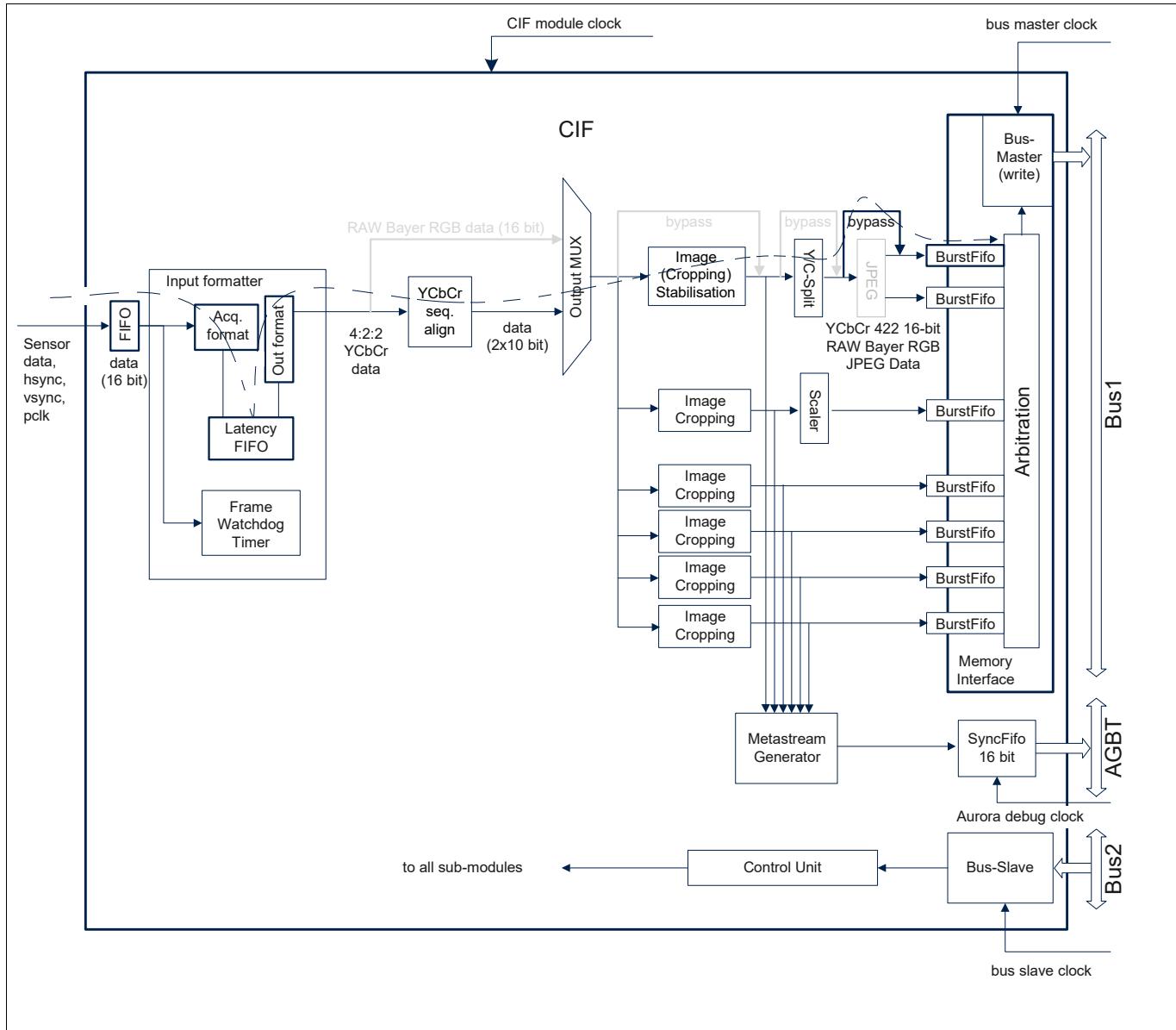
At the memory interface the Y buffer must be defined (MI\_MP\_Y\_BASE\_ADR, MI\_MP\_Y\_SIZE) and the RAW data port must be enabled (raw\_data\_en in MI\_CTRL).

After transfer being indicated by the “frame end” interrupt of the memory interface the number of transferred bytes can be read from MI\_BYTE\_CNT.

## Camera and ADC Interface (CIF)

### 26.3.14.8.2 Viewfinder Mode

This use case is to be used for displaying a video stream at an attached LCD. The CIF is responsible for capturing the video data at the sensor interface and transferring a continuous image data stream to the system memory.



**Figure 373 Active Blocks for Use Case Viewfinder**

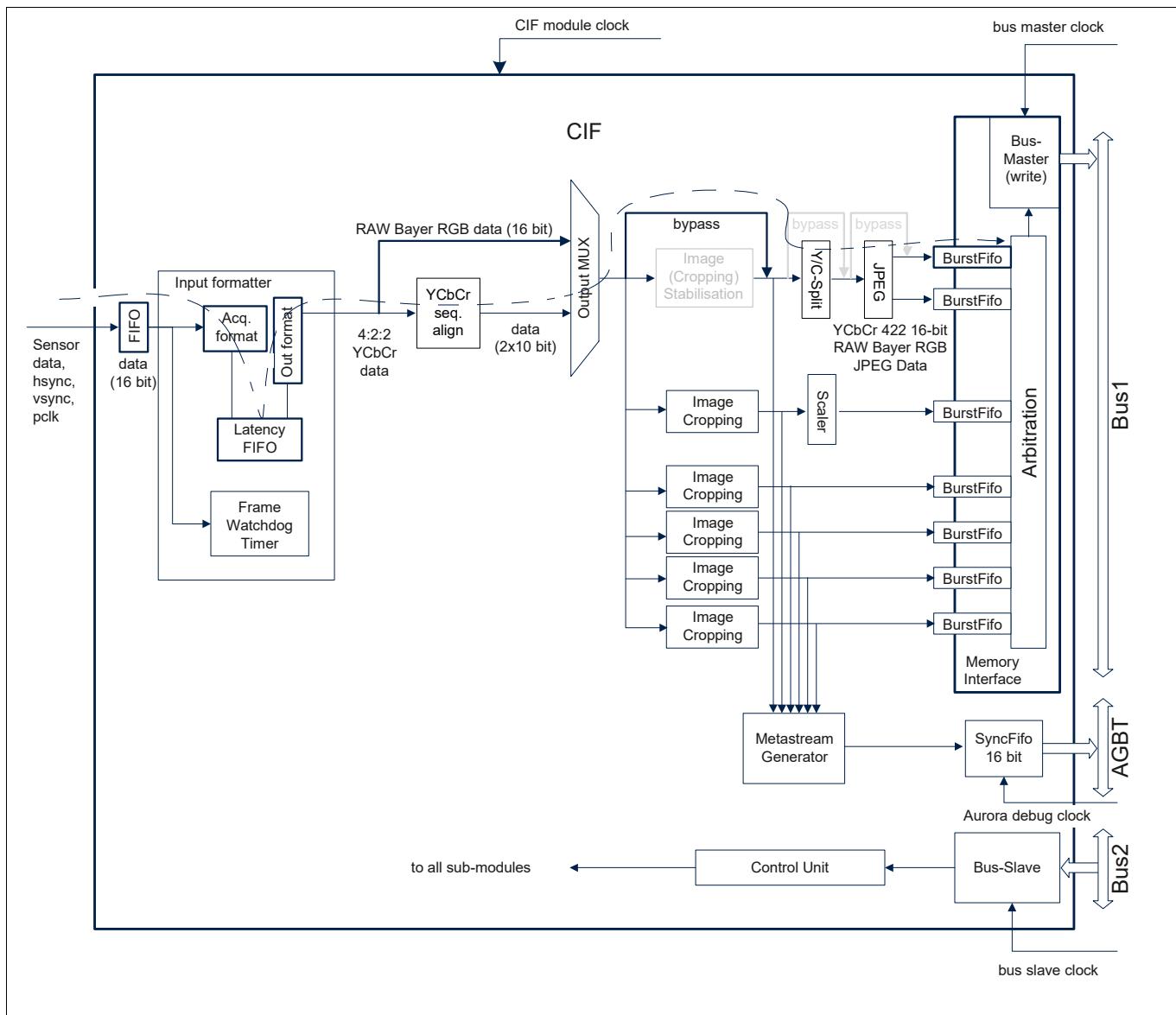
Image data are captured at the ISP. The mode depends on the camera sensor being used and can be YCbCr 4:2:2 with embedded timing control signals or external synchronization signals (selected by the `isp_mode` field in `ISP_CTRL`).

## Camera and ADC Interface (CIF)

### 26.3.14.8.3 Still Image Capture

This use case is meant for capturing a still image.

The use case viewfinder always precedes the capturing use case.



**Figure 374 Active Blocks for Use Case Still Image Capture**

In this use case the ISP samples the image and color processing can be done. The data path must be set to main 8bit luminance and chrominance processing format.

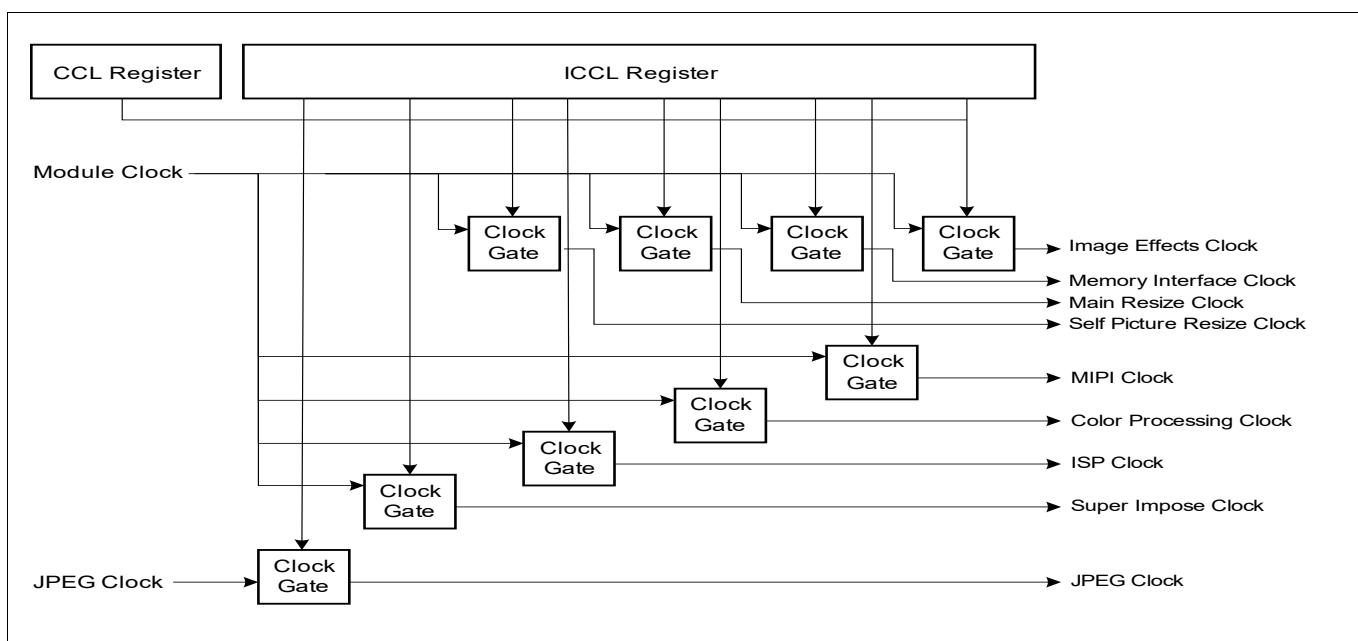
## Camera and ADC Interface (CIF)

### 26.3.14.9 Power Management

There are two main power modes for the CIF: running and switched-off. In running mode the master clock provided at the Bus interface is passed to the CIF blocks. In switched-off mode the master clock is disabled (gated) in the CIF control unit. So only register accesses are possible, but no processing. Controlling the main power modes is done using the VI\_CCL configuration register.

In running mode a further method exists to minimize power consumption of the CIF. Separate clock gating exists for all blocks inside the CIF (except the Y/C-splitter). This is controlled by the VI\_ICCL configuration register. For each module that is not needed for the actual processing (bypassed or disabled) the module clock can be disabled by clock gating as shown in Figure “[Clock Gating Scheme for the CIF](#)” on Page 51.

Clock gating for static configuration registers is handled automatically. Their clocks are enabled during read or write accesses via the Bus slave interface only, otherwise they are switched off.



**Figure 375 Clock Gating Scheme for the CIF**

### 26.3.14.10 Basics on Configuration Access

Configuration accesses are done by using the Bus slave interface. Single beat transactions with a data width of 32 bits are supported.

Each CIF sub-module contains its own local configuration registers. For optimized power consumption these registers are clocked during read/write data transfers only. This clock gating is handled automatically by the control unit and is fully transparent to the application.

---

**Camera and ADC Interface (CIF)**

## 26.4 Registers

The following section describes the CIF registers on bit level.

## Camera and ADC Interface (CIF)

### 26.4.1 CIF Control Registers

The communication and control flow to and from the CIF module is realized via internal registers. Access to the registers will be provided via an BBB slave interface. Each access is 32 bit aligned.

The CIF uses a distributed configuration register scheme. So there is no central unit containing all programming registers, but all CIF sub-modules contain their own programming registers.

**Note:** *Write accesses to reserved access locations are ignored. Read access deliver all zeros. No traps are triggered. Accesses of bus masters not enabled with the ACCENx registers cause acknowledge error.*

The register set is divided into the following register types:

- Control registers
- Configuration setting registers
- Shadow configuration setting registers
- Interrupt registers

Shadow registers are used to support dynamic data path re-programming. The next programming values are written into the configuration setting registers of a local configuration unit of a CIF internal block. Shadow registers are used for the processing part of the sub-modules to keep the current values for processing. Updating of these shadow registers can be triggered by either the configuration update bits in the sub-module control registers in terms of immediate update, or by triggering the “generate configuration update” bit in the ISP control register to automatically update all shadow registers in the whole data processing pipeline after the last pixel of a processed frame.

**Table 1179 Register Overview - CIF (ascending Offset Address)**

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
BBB_CLC	Clock Control Register	0000 <sub>H</sub>	U,SV,32	SV,E,P,32	Application Reset	<a href="#">24</a>
BBB_MODID	Module Identification Register	0004 <sub>H</sub>	SV,32	BE,32	Application Reset	<a href="#">24</a>
BBB_GPCTL	General Purpose Control Register	0008 <sub>H</sub>	U,SV,32	SV,E,P,32	Application Reset	<a href="#">25</a>
BBB_ACCENO	Access Enable Register 0	000C <sub>H</sub>	U,SV,32	SV,SE,32	Application Reset	<a href="#">25</a>
BBB_ACCEN1	Access Enable Register 1	0010 <sub>H</sub>	U,SV,32	SV,SE,32	Application Reset	<a href="#">26</a>
BBB_KRST0	Kernel Reset Register 0	0014 <sub>H</sub>	U,SV,32	SV,E,P,32	Application Reset	<a href="#">26</a>
BBB_KRST1	Kernel Reset Register 1	0018 <sub>H</sub>	U,SV,32	SV,E,P,32	Application Reset	<a href="#">27</a>
BBB_KRSTCLR	Kernel Reset Status Clear Register	001C <sub>H</sub>	U,SV,32	SV,E,P,32	Application Reset	<a href="#">28</a>
CCL	Clock Control Register	0100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">64</a>
ID	CIF Revision Identification Register	0108 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">65</a>

**Camera and ADC Interface (CIF)****Table 1179 Register Overview - CIF (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
ICCL	CIF Internal Clock Control Register	0110 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">66</a>
IRCL	CIF Internal Reset Control Register	0114 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">67</a>
DPCL	CIF Data Path Control Register	0118 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">68</a>
ISP_CTRL	ISP Global Control Register	0500 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">69</a>
ISP_ACQ_PROP	ISP Acquisition Properties Register	0504 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">71</a>
ISP_ACQ_H_OFFSET	ISP Acquisition Horizontal Offset Register	0508 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">72</a>
ISP_ACQ_V_OFFSET	ISP Acquisition Vertical Offset Register	050C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">73</a>
ISP_ACQ_H_SIZE	ISP Acquisition Horizontal Size Register	0510 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">73</a>
ISP_ACQ_V_SIZE	ISP Acquisition Vertical Size Register	0514 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">74</a>
ISP_ACQ_NR_FRAMES	ISP Acquisition Number of Frames Register	0518 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">74</a>
ISP_OUT_H_OFFSET	ISP Output Window Horizontal Offset Register	0694 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">75</a>
ISP_OUT_V_OFFSET	ISP Output Window Vertical Offset Register	0698 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">75</a>
ISP_OUT_H_SIZE	ISP Output Horizontal Picture Size Register	069C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">76</a>
ISP_OUT_V_SIZE	ISP Output Vertical Picture Size Register	06A0 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">76</a>
ISP_FLAGS_SHD	ISP Shadow Flags Register	06A8 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">77</a>
ISP_OUT_H_OFFSET_SHD	ISP Output Window Horizontal Offset Shadow Register	06AC <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">77</a>
ISP_OUT_V_OFFSET_SHD	ISP Output Window Vertical Offset Shadow Register	06B0 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">78</a>
ISP_OUT_H_SIZE_SHD	ISP Output Horizontal Picture Size Shadow Register	06B4 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">78</a>
ISP_OUT_V_SIZE_SHD	ISP Output Vertical Picture Size Shadow Register	06B8 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">79</a>

**Camera and ADC Interface (CIF)****Table 1179 Register Overview - CIF (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
ISP_IMSC	ISP Interrupt Mask Register	06BC <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">80</a>
ISP_RIS	ISP Raw Interrupt Status Register	06C0 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">81</a>
ISP_MIS	ISP Masked Interrupt Status Register	06C4 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">82</a>
ISP_ICR	ISP Interrupt Clear Register	06C8 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">83</a>
ISP_ISR	ISP Interrupt Set Register	06CC <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">84</a>
ISP_ERR	ISP Error Register	073C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">85</a>
ISP_ERR_CLR	ISP Error Clear Register	0740 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">86</a>
ISP_FRAME_COUNT	ISP Frame Counter Register	0744 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">86</a>
MI_CTRL	Memory Interface Global Control Register	1500 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">90</a>
MI_INIT	Memory Interface Control Register For Address Init And Skip Function Register	1504 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">92</a>
MI_MP_Y_BASE_ADDRESS_INIT	Memory Interface Base Address For Main Picture Y Component, JPEG or RAW Data Register	1508 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">93</a>
MI_MP_Y_SIZE_INIT	Memory Interface Size of main picture Y component, JPEG or RAW data Register	150C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">94</a>
MI_MP_Y_OFFSET_CNT_INIT	Memory Interface Offset Counter Init Value For Main Picture Y, JPEG or RAW Data Register	1510 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">95</a>
MI_MP_Y_OFFSET_CNT_START	Memory Interface Offset Counter Start Value For Main Picture Y, JPEG or RAW Data Register	1514 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">95</a>
MI_MP_Y_IRQ_OFFSETS_INIT	Memory Interface Fill Level Interrupt Offset Value For Main Picture Y, JPEG or RAW Data Register	1518 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">96</a>

**Camera and ADC Interface (CIF)****Table 1179 Register Overview - CIF (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
MI_MP_CB_BASE_AD_INIT	Memory Interface Base Address For Main Picture Cb Component Ring Buffer Register	151C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	97
MI_MP_CB_SIZE_INIT	Memory Interface Size Of Main Picture Cb Component Ring Buffer Register	1520 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	97
MI_MP_CB_OFFSETS_CNT_INIT	Memory Interface Offset Counter Init Value For Main Picture Cb Component Ring Buffer Register	1524 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	98
MI_MP_CB_OFFSETS_CNT_START	Memory Interface Offset Counter Start Value For Main Picture Cb Component Ring Buffer Register	1528 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	98
MI_MP_CR_BASE_AD_INIT	Memory Interface Base Address For Main Picture Cr Component Ring Buffer Register	152C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	99
MI_MP_CR_SIZE_INIT	Memory Interface Size Of Main Picture Cr Component Ring Buffer Register	1530 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	100
MI_MP_CR_OFFSETS_CNT_INIT	Memory Interface Offset Counter Init value For Main Picture Cr Component Ring Buffer Register	1534 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	100
MI_MP_CR_OFFSETS_CNT_START	Memory Interface Offset Counter Start Value For Main Picture Cr Component Ring Buffer Register	1538 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	101
MI_BYTE_CNT	Memory Interface Counter Value of JPEG or RAW Data Bytes Register	1570 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	102
MI_CTRL_SHD	Memory Interface Global Control Internal Shadow Register	1574 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	103
MI_MP_Y_BASE_AD_SHD	Memory Interface Base Address Shadow Register For Main Picture Y Component, JPEG Register	1578 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	104

**Camera and ADC Interface (CIF)****Table 1179 Register Overview - CIF (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
MI_MP_Y_SIZE_SHD	Memory Interface Size Shadow Register of Main Picture Y Component,JPEG or RAW Data Register	157C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>104</b>
MI_MP_Y_OFFSET_CNT_SHD	Memory Interface Current Offset Counter of Main Picture Y Component JPEG or RAW Register	1580 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>105</b>
MI_MP_Y_IRQ_OFFS_FFS_SHD	Memory Interface Shadow Register of Fill Level Interrupt Offset Value For Main Picture Y Register	1584 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>105</b>
MI_MP_CB_BASE_AD_SHD	Memory Interface Base Address Shadow Register For Main Picture Cb Component Ring Register	1588 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>106</b>
MI_MP_CB_SIZE_SHD	Memory Interface Size Shadow Register Of Main Picture Cb Component Ring Buffer Register	158C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>106</b>
MI_MP_CB_OFFSET_CNT_SHD	Memory Interface Current Offset Counter Of Main Picture Cb Component Ring Buffer Register	1590 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>107</b>
MI_MP_CR_BASE_AD_SHD	Memory Interface Base Address Shadow Register For Main Picture Cr Component Ring Register	1594 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>107</b>
MI_MP_CR_SIZE_SHD	Memory Interface Size Shadow Register Of Main Picture Cr Component Ring Buffer Register	1598 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>108</b>
MI_MP_CR_OFFSET_CNT_SHD	Memory Interface Current Offset Counter Of Main Picture Cr Component Ring Buffer Register	159C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>108</b>
MI_IMSC	MI Interrupt Mask '1' interrupt active '0' interrupt masked	15F8 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>109</b>
MI_RIS	MI Raw Interrupt Status Register	15FC <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>110</b>
MI_MIS	MI Masked Interrupt Status Register	1600 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>111</b>

**Camera and ADC Interface (CIF)****Table 1179 Register Overview - CIF (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
MI_ICR	MI Interrupt Clear Register	1604 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">112</a>
MI_ISR	MI Interrupt Set Register	1608 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">113</a>
MI_STATUS	MI Status Register	160C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">114</a>
MI_STATUS_CLR	MI Status Clear Register	1610 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">114</a>
JPE_GEN_HEADER	JPE Command To Start Stream Header Generation Register	1900 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">116</a>
JPE_ENCODE	JPE Start Command To Start JFIF Stream Encoding Register	1904 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">116</a>
JPE_INIT	JPE Automatic Configuration Update Register	1908 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">117</a>
JPE_Y_SCALE_EN	JPE Y Value Scaling Control Register	190C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">117</a>
JPE_CBCR_SCAL_E_EN	JPE Cb/Cr Value Scaling Control Register	1910 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">118</a>
JPE_TABLE_FLUSH	JPE Header Generation Debug Register	1914 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">118</a>
JPE_ENC_HSIZE	JPEG Codec Horizontal Image Size For Encoding Register	1918 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">119</a>
JPE_ENC_VSIZE	JPEG Codec Vertical Image Size For Encoding Register	191C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">119</a>
JPE_PIC_FORMAT	JPEG Picture Encoding Format Register	1920 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">120</a>
JPE_RESTART_INTERVAL	JPE Restart Marker Insertion Register	1924 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">120</a>
JPE_TQ_Y_SELECT	Q-table Selector 0, Quant. Table For Y Component	1928 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">121</a>
JPE_TQ_U_SELECT	Q-table Selector 1, Quant. Table For U Component	192C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">121</a>
JPE_TQ_V_SELECT	Q-table Selector 2 Quant Table For V Component	1930 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">122</a>
JPE_DC_TABLE_SELECT	JPE Huffman Table Selector For DC Values Register	1934 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">122</a>

**Camera and ADC Interface (CIF)****Table 1179 Register Overview - CIF (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
JPE_AC_TABLE_SELECT	JPE Huffman Table Selector For AC Values Register	1938 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">123</a>
JPE_TABLE_DATA	JPE Table Programming Register	193C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">123</a>
JPE_TABLE_ID	JPE Table Programming Select Register	1940 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">124</a>
JPE_TAC0_LEN	JPE Huffman AC Table 0 Length Register	1944 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">124</a>
JPE_TDC0_LEN	JPE Huffman DC Table 0 Length Register	1948 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">125</a>
JPE_TAC1_LEN	JPE Huffman AC Table 1 Length Register	194C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">125</a>
JPE_TDC1_LEN	JPE Huffman DC Table 1 Length Register	1950 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">126</a>
JPE_ENCODER_BSY	JPE Encoder Status Flag Register	1958 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">126</a>
JPE_HEADER_MODE	JPE Header Mode Definition Register	195C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">127</a>
JPE_ENCODE_MODE	JPE Encode Mode Register	1960 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">127</a>
JPE_DEBUG	JPE Debug Information Register	1964 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">128</a>
JPE_ERROR_IMR	JPE Error Interrupt Mask Register	1968 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">129</a>
JPE_ERROR_RIS	JPE Error Raw Interrupt Status Register	196C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">129</a>
JPE_ERROR_MIS	JPE Error Masked Interrupt Status Register	1970 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">130</a>
JPE_ERROR_ICR	JPE Error Interrupt Clear Register	1974 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">131</a>
JPE_ERROR_ISR	JPE Error Interrupt Set Register	1978 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">131</a>
JPE_STATUS_IMR	JPEG Status Interrupt Mask Register	197C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">132</a>
JPE_STATUS_RIS	JPEG Status Raw Interrupt Status Register	1980 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">133</a>
JPE_STATUS_MIS	JPEG Status Masked Interrupt Status Register	1984 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">133</a>
JPE_STATUS_ICR	JPEG Status Interrupt Clear Register	1988 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">134</a>

**Camera and ADC Interface (CIF)****Table 1179 Register Overview - CIF (ascending Offset Address) (cont'd)**

<b>Short Name</b>	<b>Long Name</b>	<b>Offset Address</b>	<b>Access Mode</b>		<b>Reset</b>	<b>Page Number</b>
			<b>Read</b>	<b>Write</b>		
JPE_STATUS_ISR	JPEG Status Interrupt Set Register	198C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>134</b>
ISPIS_CTRL	ISP Image Stabilization Control Register	2400 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>141</b>
ISPIS_RECENTER	ISP Image Stabilization Recenter Register	2404 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>141</b>
ISPIS_H_OFFSETS	ISP Image Stabilization Horizontal Offset Of Output Window Register	2408 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>142</b>
ISPIS_V_OFFSETS	ISP Image Stabilization Vertical Offset Of Output Window Register	240C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>142</b>
ISPIS_H_SIZE	ISP Image Stabilization Output Horizontal Picture Size Register	2410 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>143</b>
ISPIS_V_SIZE	ISP Image Stabilization Output Vertical Picture Size Register	2414 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>143</b>
ISPIS_MAX_DX	ISP Image Stabilization Maximum Horizontal Displacement Register	2418 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>144</b>
ISPIS_MAX_DY	ISP Image Stabilization Maximum Vertical Displacement Register	241C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>144</b>
ISPIS_DISPLACE	ISP Image Stabilization Camera Displacement Register	2420 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>145</b>
ISPIS_H_OFFSETS_SHADOW	ISP Image Current Horizontal Offset Of Output Window Shadow Register	2424 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>145</b>
ISPIS_V_OFFSETS_SHADOW	ISP Image Current Vertical Offset Of Output Window Shadow Register	2428 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>146</b>
ISPIS_H_SIZE_SHADOW	ISP Image Current Output Horizontal Picture Size Shadow Register	242C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>146</b>
ISPIS_V_SIZE_SHADOW	ISP Image Current Output Vertical Picture Size Shadow Register	2430 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>147</b>
WD_CTRL	Watchdog Control Register	2500 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<b>135</b>

## Camera and ADC Interface (CIF)

Table 1179 Register Overview - CIF (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
WD_V_TIMEOUT	Watchdog Vertical Timeout Register	2504 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">136</a>
WD_H_TIMEOUT	Watchdog Horizontal Timeout Register	2508 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">136</a>
WD_IMSC	Watchdog Interrupt Mask Register	250C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">137</a>
WD_RIS	Watchdog Raw Interrupt Status Register	2510 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">137</a>
WD_MIS	Watchdog Masked Interrupt Status Register	2514 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">138</a>
WD_ICR	Watchdog Interrupt Clear Register	2518 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">139</a>
WD_ISR	Watchdog Interrupt Set Register	251C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">139</a>
LDS_CTRL	Linear Downscaler Control Register	2600 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">88</a>
LDS_FAC	Linear Downscaler Factor Register	2604 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">89</a>
DP_CTRL	Debug Path Control Register	2800 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">174</a>
DP_PDIV_CTRL	Debug Path Predivider Control Register	2804 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">175</a>
DP_FLC_STAT	Debug Path Frame/Line Counter Status Register	2808 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">176</a>
DP_PDIV_STAT	Debug Path Predivider Counter Status Register	280C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">176</a>
DP_TSC_STAT	Debug Path Timestamp Counter Status Register	2810 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">177</a>
DP_UDS_X	Debug Path User Defined Symbol x Register	2814 <sub>H</sub> +x *4	U,SV,32	U,SV,P,32	Application Reset	<a href="#">177</a>
EP_i_IC_CTRL	Extra Path i Image Cropping Control Register	2A00 <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">167</a>
EP_i_IC_RECENT_ER	Extra Path i Image Cropping Recenter Register	2A04 <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">167</a>
EP_i_IC_H_OFFSETS	Extra Path i Image Cropping Horizontal Offset of Output Window Register	2A08 <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">168</a>
EP_i_IC_V_OFFSETS	Extra Path i Image Cropping Vertical Offset Of Output Window Register	2A0C <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">168</a>

## Camera and ADC Interface (CIF)

Table 1179 Register Overview - CIF (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
EP_i_IC_H_SIZE	Extra Path i Image Cropping Output Horizontal Picture Size Register	2A10 <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">169</a>
EP_i_IC_V_SIZE	Extra Path i Image Cropping Output Vertical Picture Size Register	2A14 <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">169</a>
EP_i_IC_MAX_DX	Extra Path i Image Cropping Maximum Horizontal Displacement Register	2A18 <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">170</a>
EP_i_IC_MAX_DY	Extra Path i Image Cropping Maximum Vertical Displacement Register	2A1C <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">170</a>
EP_i_IC_DISPLAC E	Extra Path i Image Cropping Camera Displacement Register	2A20 <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">171</a>
EP_i_IC_H_OFFSETS _SHD	Extra Path i Image Cropping Current Horizontal Offset of Output Window Shadow Register	2A24 <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">172</a>
EP_i_IC_V_OFFSETS _SHD	Extra Path i Image Cropping Current Vertical Offset Of Output Window Shadow Register	2A28 <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">172</a>
EP_i_IC_H_SIZE_ SHD	Extra Path i Image Cropping Current Output Horizontal Picture Size Shadow Register	2A2C <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">173</a>
EP_i_IC_V_SIZE_ SHD	Extra Path i Image Cropping Current Output Vertical Picture Size Shadow Register	2A30 <sub>H</sub> +i* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">173</a>
MIEP_STA_ERR	Extra Path Error Register	3500 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">148</a>
MIEP_STA_ERR_ CLR	Extra Path Status Error Clear Register	3504 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">149</a>
MIEP_IMSC	MI Extra Path Interrupt Mask '1': interrupt active, '0': interrupt masked	3508 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">150</a>
MIEP_RIS	MI Extra Path Raw Interrupt Status Register	350C <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">151</a>
MIEP_MIS	MI Extra Path Masked Interrupt Status Register	3510 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">153</a>

## Camera and ADC Interface (CIF)

Table 1179 Register Overview - CIF (ascending Offset Address) (cont'd)

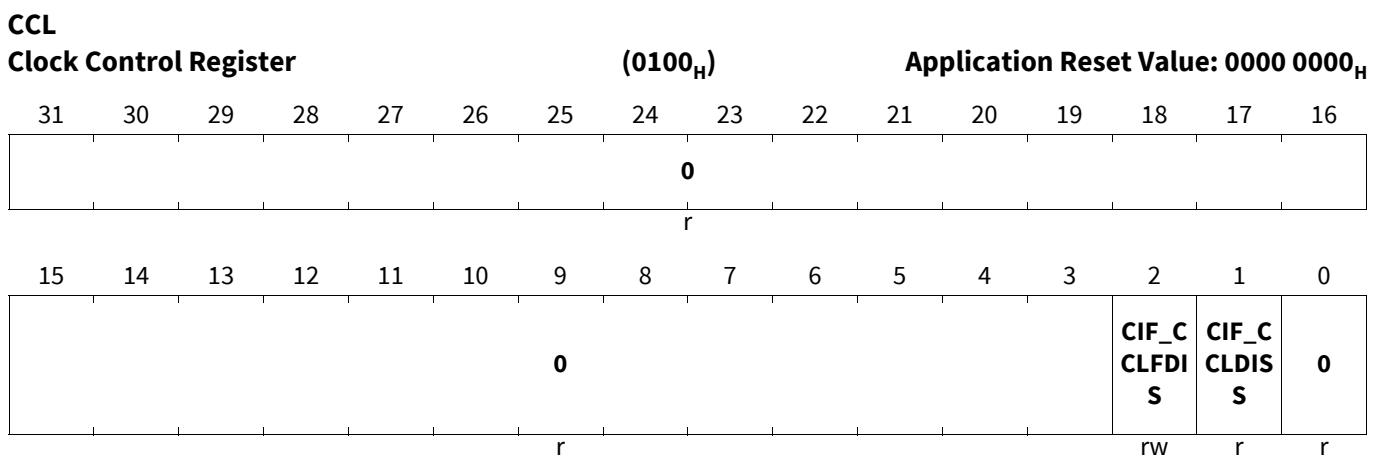
Short Name	Long Name	Offset Address	Access Mode		Reset	Page Number
			Read	Write		
MIEP_ICR	MI Extra Path Interrupt Clear Register	3514 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">154</a>
MIEP_ISR	MI Extra Path Interrupt Set Register	3518 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">156</a>
MIEP_j_CTRL	Memory Interface Extra Path j Control Register	3600 <sub>H</sub> +j* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">157</a>
MIEP_j_INIT	Memory Interface Extra Path j Control Register For Address Init And Skip Function Register	3604 <sub>H</sub> +j* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">159</a>
MIEP_j_BASE_AD_INIT	Memory Interface Base Address for Extra Path j Data Buffer Register	3608 <sub>H</sub> +j* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">160</a>
MIEP_j_SIZE_INIT	Memory Interface Size of Extra Path j Data Buffer Register	360C <sub>H</sub> +j* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">160</a>
MIEP_j_OFFSETS_CNT_INIT	Memory Interface Offset Counter Init Value For Extra Path j Buffer Register	3610 <sub>H</sub> +j* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">161</a>
MIEP_j_OFFSETS_CNT_START	Memory Interface Offset Counter Start Value for Extra Path j Register	3614 <sub>H</sub> +j* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">162</a>
MIEP_j_IRQ_OFFSETS_INIT	Memory Interface Fill Level Interrupt Offset Value For Extra Path Data Register	3618 <sub>H</sub> +j* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">162</a>
MIEP_j_CTRL_SHD	Memory Interface Extra Path j Control Internal Shadow Register	361C <sub>H</sub> +j* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">164</a>
MIEP_j_BASE_ADDR_SHD	Memory Interface Base Address Shadow Register for Extra Path j Buffer Register	3620 <sub>H</sub> +j* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">164</a>
MIEP_j_SIZE_SHD	Memory Interface Size Shadow Register of Extra Path j Buffer Register	3624 <sub>H</sub> +j* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">165</a>
MIEP_j_OFFSETS_CNT_SHD	Memory Interface Current Offset Counter of Extra Path j Buffer Register	3628 <sub>H</sub> +j* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">165</a>
MIEP_j_IRQ_OFFSETS_SHD	Memory Interface Shadow Register of Fill Level Interrupt Offset Value For Extra Path j Register	362C <sub>H</sub> +j* 100 <sub>H</sub>	U,SV,32	U,SV,P,32	Application Reset	<a href="#">166</a>

**Camera and ADC Interface (CIF)**

The registers are addressed wordwise.

**Table 1180 Registers Access Types**

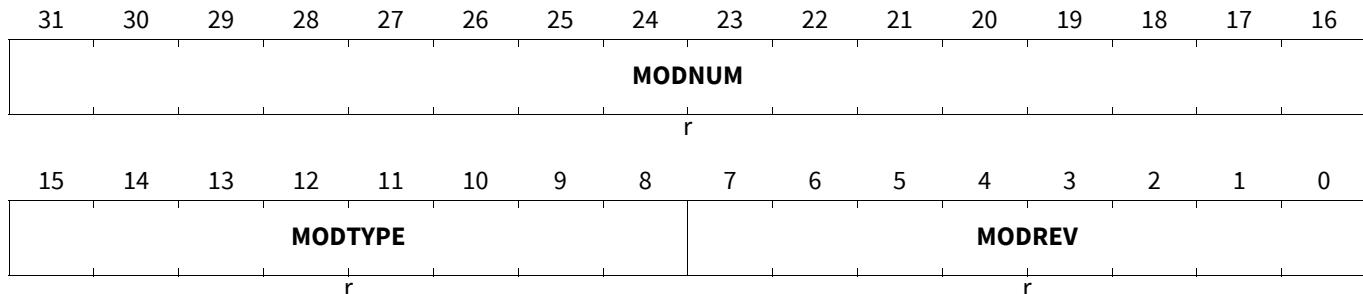
Mode	Symbol	Description Hardware (HW)	Description Software (SW)
<b>Basic Access Types</b>			
read/write	rw	Register is used as input for the HW	Register is read and writable by SW
read	r	Register is written by HW (register between input and output -> one cycle delay)	Value written by SW is ignored by HW; that is, SW may write any value to this field without affecting HW behavior
write	w	Register is written by software and affects hardware behavior with every write by software.	Register is writable by SW. When read, the register does not return the value that has been written previously, but some constant value instead.

**26.4.1.1 CIF Clock Control Registers****Clock Control Register**

Field	Bits	Type	Description
<b>CIF_CCLDISS</b>	1	r	<b>Status of cif_ccl[2] bit</b> (copy of cif_ccl[2])
<b>CIF_CCLFDIS</b>	2	rw	<b>Clock Control Logic disable</b> 0 <sub>B</sub> processing/cfg-clocks for all CIF sub modules enabled 1 <sub>B</sub> processing/cfg-clocks for all CIF sub modules disabled w/o access to ID and CIF_CCL register
<b>0</b>	0, 31:3	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****26.4.1.2 CIF Custom Registers****CIF Revision Identification Register**

**ID**  
**CIF Revision Identification Register** **(0108<sub>H</sub>)** **Application Reset Value: 00B3 C002<sub>H</sub>**



Field	Bits	Type	Description
<b>MODREV</b>	7:0	r	<b>Module Revision Number</b> This bit field defines the module revision number.02
<b>MODTYPE</b>	15:8	r	<b>Module Type</b> This bit field defines the module as a 32-bit module.C0
<b>MODNUM</b>	31:16	r	<b>Module Number Value</b> This bit field defines the module as a CIF.B3

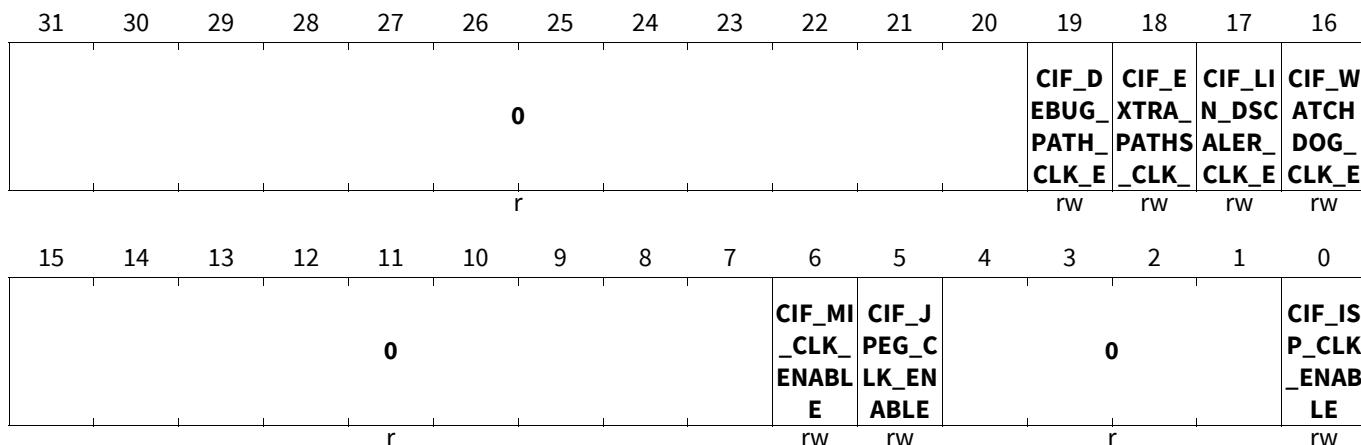
## Camera and ADC Interface (CIF)

## 26.4.1.3 CIF Internal Control Registers

## CIF Internal Clock Control Register

ICCL

## CIF Internal Clock Control Register

(0110<sub>H</sub>)Application Reset Value: 000F 0061<sub>H</sub>

Field	Bits	Type	Description
<b>CIF_ISP_CLK_ENABLE</b>	0	rw	<b>ISP processing clock enable</b> 0 <sub>B</sub> power safe 1 <sub>B</sub> processing mode
<b>CIF_JPEG_CLK_ENABLE</b>	5	rw	<b>JPEG encoder clock enable</b> 0 <sub>B</sub> power safe 1 <sub>B</sub> processing mode
<b>CIF_MI_CLK_ENABLE</b>	6	rw	<b>Memory interface clock enable</b> 0 <sub>B</sub> power safe 1 <sub>B</sub> processing mode
<b>CIF_WATCHDOG_CLK_ENABLE</b>	16	rw	<b>Security Watchdog clock enable</b> 0 <sub>B</sub> power safe 1 <sub>B</sub> processing mode
<b>CIF_LINEAR_DSCALER_CLK_ENABLE</b>	17	rw	<b>Linear Downscaler clock enable</b> 0 <sub>B</sub> power safe 1 <sub>B</sub> processing mode
<b>CIF_EXTRA_PATHS_CLK_ENABLE</b>	18	rw	<b>Extra Paths clock enable</b> 0 <sub>B</sub> power safe 1 <sub>B</sub> processing mode
<b>CIF_DEBUG_PATH_CLK_ENABLE</b>	19	rw	<b>Debug Path clock enable</b> 0 <sub>B</sub> power safe 1 <sub>B</sub> processing mode
<b>0</b>	4:1, 15:7, 31:20	r	<b>Reserved</b> Read as 0, should be written with 0.

## Camera and ADC Interface (CIF)

## CIF Internal Reset Control Register

IRCL

CIF Internal Reset Control Register

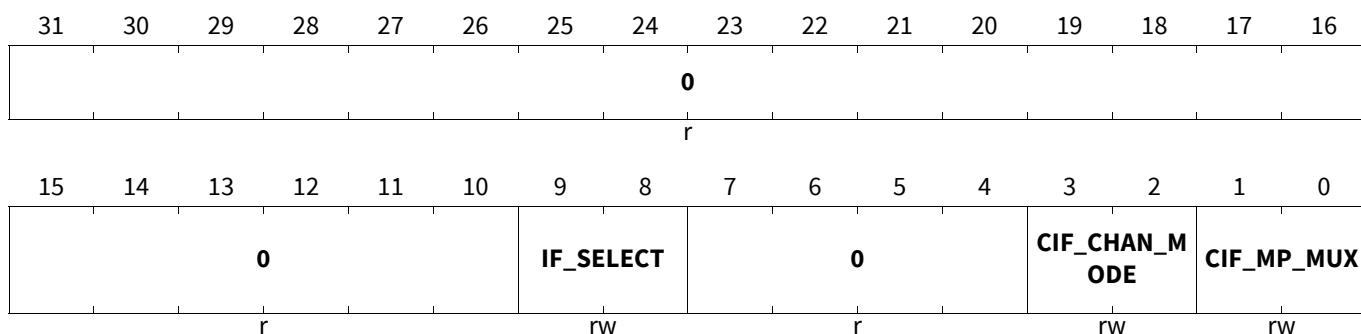
(0114<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
												CIF_D EBUG_ PATH_	CIF_E XTRA_	CIF_LI N_DSC	CIF_W ATCH_
												RST	RST	ALER_	DOG_
												r	rw	rw	rw
0															
												CIF_G LOBAL_	CIF_MI SOFT_	CIF_J PEG_S	CIF_Y CS_SO
												RST	RST	OFTR	FT_RS
												r	0	0	0
												rw	rw	r	rw
												r	r	r	r

Field	Bits	Type	Description
<b>CIF_ISP_SOFT_RST</b>	0	rw	<b>Isp software reset</b> 0 <sub>B</sub> processing mode 1 <sub>B</sub> reset state
<b>CIF_YCS_SOF_T_RST</b>	2	rw	<b>Y/C splitter software reset</b> 0 <sub>B</sub> processing mode 1 <sub>B</sub> reset state
<b>CIF_JPEG_SOFT_RST</b>	5	rw	<b>JPEG encoder software reset</b> 0 <sub>B</sub> processing mode 1 <sub>B</sub> reset state
<b>CIF_MI_SOFT_RST</b>	6	rw	<b>Memory interface software reset</b> 0 <sub>B</sub> processing mode 1 <sub>B</sub> reset state
<b>CIF_GLOBAL_RST</b>	7	rw	<b>Soft reset of entire CIF</b> 0 <sub>B</sub> processing mode 1 <sub>B</sub> reset state
<b>CIF_WATCHDOG_RST</b>	16	rw	<b>Securitiy Watchdog software reset</b> 0 <sub>B</sub> processing mode 1 <sub>B</sub> reset state
<b>CIF_LIN_DSCALER_RST</b>	17	rw	<b>Linear Downscaler software reset</b> 0 <sub>B</sub> processing mode 1 <sub>B</sub> reset state
<b>CIF_EXTRA_PATHS_RST</b>	18	rw	<b>Extra Paths software reset</b> 0 <sub>B</sub> processing mode 1 <sub>B</sub> reset state
<b>CIF_DEBUG_PATH_RST</b>	19	rw	<b>Debug Path software reset</b> 0 <sub>B</sub> processing mode 1 <sub>B</sub> reset state

**Camera and ADC Interface (CIF)**

Field	Bits	Type	Description
0	1, 4:3, 15:8, 31:20	r	<b>Reserved</b> Read as 0, should be written with 0.

**CIF Data Path Control Register****DPCL****CIF Data Path Control Register****(0118<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>CIF_MP_MUX</b>	1:0	rw	<b>Data path selector for main path</b> Other values are reserved. 00 <sub>B</sub> disabled 01 <sub>B</sub> data to MI uncompressed 10 <sub>B</sub> data to JPEG encoder
<b>CIF_CHAN_M_ODE</b>	3:2	rw	<b>Y/C splitter channel mode</b> Other values are reserved. 00 <sub>B</sub> disabled 01 <sub>B</sub> main path and RAW data mode
<b>IF_SELECT</b>	9:8	rw	<b>Selects input interface</b> Other values are reserved. 00 <sub>B</sub> parallel interface
0	7:4, 31:10	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****26.4.2 ISP Programming Registers**

The address of each CIF ISP programming register is evaluated as `CIF_ISP_BASE + Offset`.

**26.4.2.1 ISP Control Registers****ISP Global Control Register****ISP\_CTRL**

**ISP Global Control Register** **(0500<sub>H</sub>)** **Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>0</b>	<b>ISP_C SM_C_ RANG E</b>	<b>ISP_C SM_Y_ RANG E</b>	<b>0</b>	<b>ISP_G EN_CF G_UP D</b>	<b>ISP_C FG_UP D</b>	<b>0</b>		<b>0</b>		<b>ISP_IN FORM _ENAB LE</b>	<b>ISP_MODE</b>		<b>ISP_E NABLE</b>		
r	rw	rw	r	w	w	r		r		rw	rw		rw		rw

Field	Bits	Type	Description
<b>ISP_ENABLE</b>	0	rw	<b>ISP output enable</b> Controls output formater frame synchronously, if <code>isp_gen_cfg_upd</code> is used to activate this bit. For immediate update <code>isp_cfg_upd</code> must be used. <code>0<sub>B</sub></code> ISP output OFF <code>1<sub>B</sub></code> ISP output ON
<b>ISP_MODE</b>	3:1	rw	<b>ISP Mode</b> Unused values are reserved. <code>000<sub>B</sub></code> RAW picture <code>001<sub>B</sub></code> ITU-R BT.656 (YUV with embedded sync) <code>010<sub>B</sub></code> ITU-R BT.601 (YUV input with H and Vsync signals) <code>100<sub>B</sub></code> datamode (ISP bypass, sync signals interpreted as data enable) <code>110<sub>B</sub></code> RAW picture mode with ITU-R BT.656 synchronization
<b>ISP_INFORM_ENABLE</b>	4	rw	<b>ISP Input Formater Enable</b> Controls input formater frame synchronously, if <code>isp_gen_cfg_upd</code> is used to activate this bit. For immediate update <code>isp_cfg_upd</code> must be used <code>0<sub>B</sub></code> Input Formater is in deactivated <code>1<sub>B</sub></code> Input Formater is active
<b>ISP_CFG_UPD</b>	9	w	<b>ISP Config Update</b> <code>0<sub>B</sub></code> no effect <code>1<sub>B</sub></code> immediately configures (update) shadow registers
<b>ISP_GEN_CFG_UPD</b>	10	w	<b>ISP Generate Config Update</b> <code>0<sub>B</sub></code> no effect <code>1<sub>B</sub></code> generate frame synchronous configuration signal at the output of ISP for shadow registers of the following processing modules

**Camera and ADC Interface (CIF)**

<b>Field</b>	<b>Bits</b>	<b>Type</b>	<b>Description</b>
<b>ISP_CSM_Y_RANGE</b>	13	rw	<b>Color Space Matrix luminance clipping range for ISP output</b> $0_B$ Y range 16..235 according to ITU-R BT.601standard $1_B$ full Y range 0..255
<b>ISP_CSM_C_RANGE</b>	14	rw	<b>Color Space Matrix chrominance clipping range for ISP output</b> $0_B$ CbCr range 16..240 according to ITU-R BT.601standard $1_B$ full UV range 0..255
<b>0</b>	8:5, 12:11, 31:15	r	<b>Reserved</b> Read as 0, should be written with 0.

## Camera and ADC Interface (CIF)

## 26.4.2.2 ISP Acquisition Registers

## ISP Acquisition Properties Register

## ISP\_ACQ\_PROP

## ISP Acquisition Properties Register

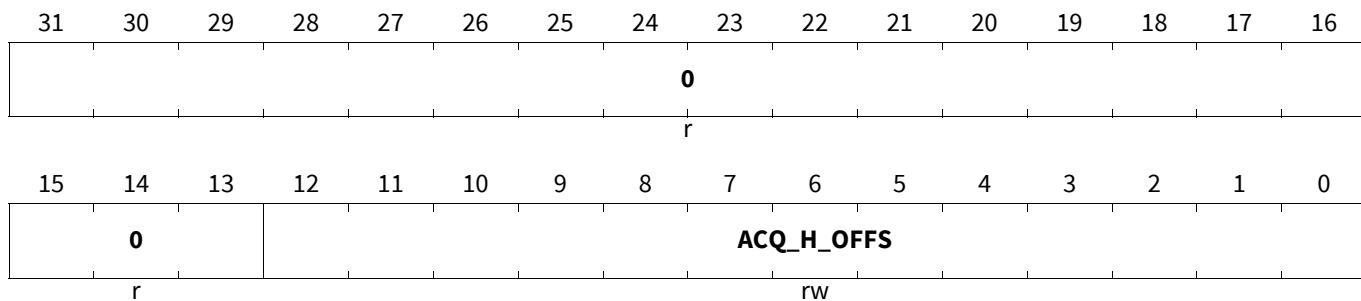
(0504<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							0				<b>INPUT_SELECTION_NO_A</b>		0		
							r				rw		r		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				<b>FIELD_INVERT</b>	<b>FIELD_SELECTION</b>		<b>CCIR_SEQ</b>				0		<b>VSYNC_POL</b>	<b>HSYNC_POL</b>	<b>SAMPLE_EDGE</b>
				rw	rw	rw	rw	rw			r		rw	rw	rw

Field	Bits	Type	Description
<b>SAMPLE_EDGE</b>	0	rw	<b>Sample Edge</b> 0 <sub>B</sub> negative edge sampling 1 <sub>B</sub> positive edge sampling
<b>HSYNC_POL</b>	1	rw	<b>Horizontal sync polarity</b> 0 <sub>B</sub> high active 1 <sub>B</sub> low active
<b>VSYNC_POL</b>	2	rw	<b>Vertical sync polarity</b> 0 <sub>B</sub> high active 1 <sub>B</sub> low active
<b>CCIR_SEQ</b>	8:7	rw	<b>CCIR Sequence</b> This bit field defines the output sequence of the Acquisition Format block. 00 <sub>B</sub> YCbYCr 01 <sub>B</sub> YCrYCb 10 <sub>B</sub> CbYCrY
<b>FIELD_SELECTION</b>	10:9	rw	<b>Field Selection</b> 00 <sub>B</sub> sample all fields (don't care about fields) 01 <sub>B</sub> sample only even fields 10 <sub>B</sub> sample only odd fields 11 <sub>B</sub> reserved
<b>FIELD_INVERT</b>	11	rw	<b>Field Invert</b> If set to 1 <sub>B</sub> the field-id will be inverted (even fields will become odd ones and vice versa).

**Camera and ADC Interface (CIF)**

Field	Bits	Type	Description
<b>INPUT_SELECTION</b>	15:12	rw	<b>Input Selection</b>  0 <sub>H</sub> 8-bit external Interface if enabled append 8 MSBs as LSBs 1 <sub>H</sub> 8-bit external Interface if enabled append 8 zeroes as LSBs 2 <sub>H</sub> 10-bit external Interface if enabled append 6 MSBs as LSBs 3 <sub>H</sub> 10-bit external Interface if enabled append 6 zeroes as LSBs 4 <sub>H</sub> 12-bit external Interface if enabled append 4 MSBs as LSBs 5 <sub>H</sub> 12-bit external Interface if enabled append 4 zeroes as LSBs 6 <sub>H</sub> 14-bit external Interface if enabled append 2 MSBs as LSBs 7 <sub>H</sub> 14-bit external Interface if enabled append 2 zeroes as LSBs 8 <sub>H</sub> 16-bit external Interface
<b>INPUT_SELECTION_NO_APPEND</b>	20	rw	<b>Input Selection No Append</b> This field controls if the input of an external Interface gets appended with zeroes or its MSBs or shifted right to be LSB aligned. 0 <sub>B</sub> append enabled 1 <sub>B</sub> append disabled
<b>0</b>	6:3, 19:16, 31:21	r	<b>Reserved</b> Read as 0, should be written with 0.

**ISP Acquisition Horizontal Offset Register****ISP\_ACQ\_H\_OFFSETS****ISP Acquisition Horizontal Offset Register**(0508<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>ACQ_H_OFFSETS</b>	12:0	rw	<b>Horizontal sample offset</b> In sensor data samples (yuv: 4 samples=2pix)
<b>0</b>	31:13	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****ISP Acquisition Vertical Offset Register****ISP\_ACQ\_V\_OFFSETS****ISP Acquisition Vertical Offset Register****(050C<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										ACQ_V_OFFSETS					
r															rw

Field	Bits	Type	Description
<b>ACQ_V_OFFSETS</b>	11:0	rw	<b>Vertical sample offset</b> In lines
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

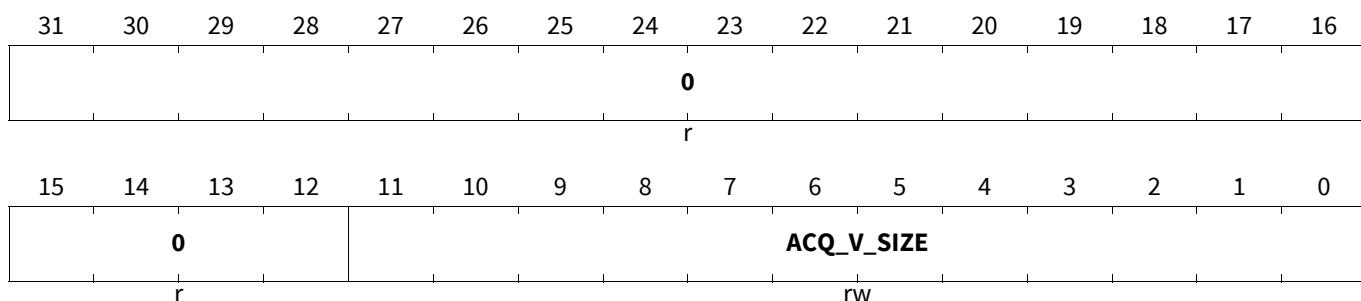
**ISP Acquisition Horizontal Size Register****ISP\_ACQ\_H\_SIZE****ISP Acquisition Horizontal Size Register****(0510<sub>H</sub>)****Application Reset Value: 0000 0A28<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										ACQ_H_SIZE					
r															rw

Field	Bits	Type	Description
<b>ACQ_H_SIZE</b>	12:0	rw	<b>Horizontal sample size</b> In sensor data samples YUV input: 2 samples=1 pixel, else 1 sample=1 pixel; So in YUV mode ACQ_H_SIZE must be twice as large as horizontal image size Horizontal image size must always be even except in RAW picture mode; if an odd size is programmed the value will be truncated to even size
<b>0</b>	31:13	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****ISP Acquisition Vertical Size Register****ISP\_ACQ\_V\_SIZE**

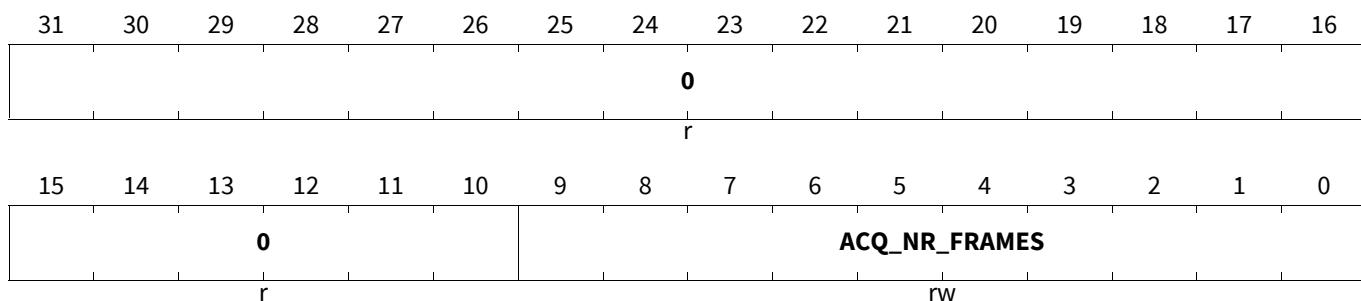
ISP Acquisition Vertical Size Register

(0514<sub>H</sub>)Application Reset Value: 0000 0800<sub>H</sub>

Field	Bits	Type	Description
ACQ_V_SIZE	11:0	rw	<b>Vertical sample size</b> In lines
0	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**ISP Acquisition Number of Frames Register****ISP\_ACQ\_NR\_FRAMES**

ISP Acquisition Number of Frames Register

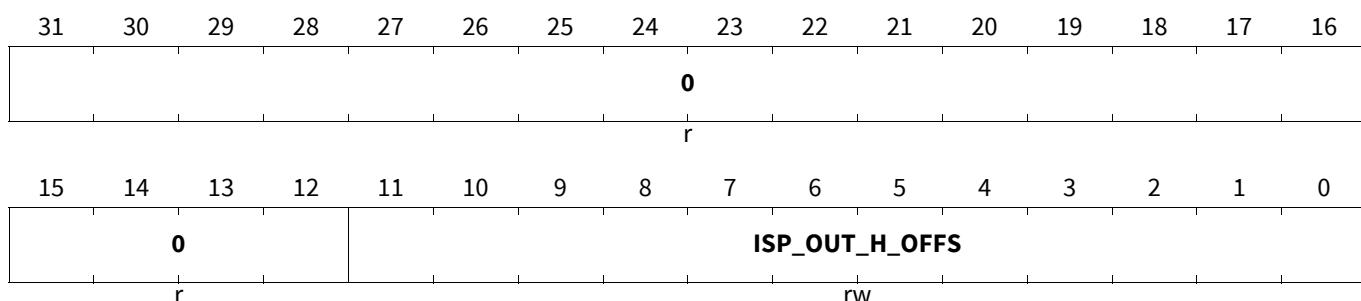
(0518<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
ACQ_NR_FRA MES	9:0	rw	<b>Number of Input Frames</b> To be sampled (0 = continuous)
0	31:10	r	<b>Reserved</b> Read as 0, should be written with 0.

## Camera and ADC Interface (CIF)

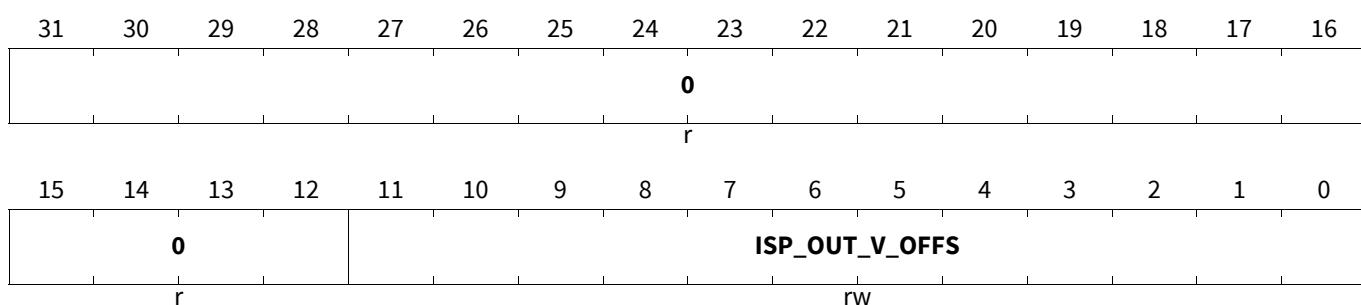
## 26.4.2.3 ISP Output Control Registers

## ISP Output Window Horizontal Offset Register

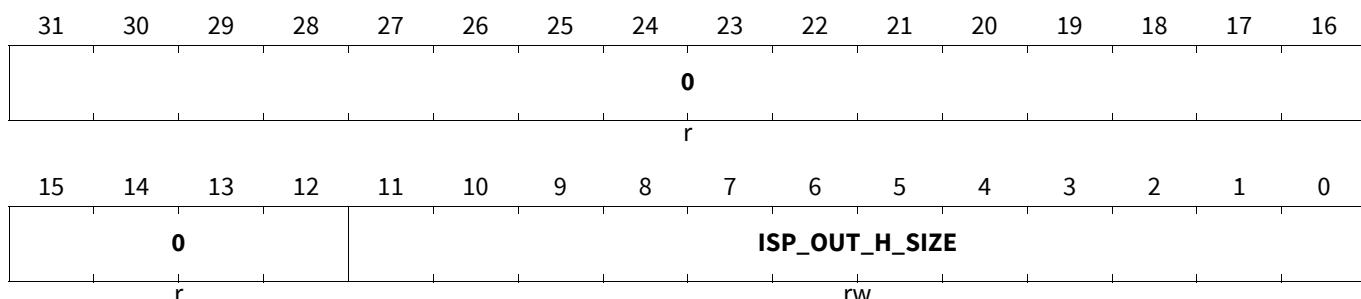
**ISP\_OUT\_H\_OFFSETS****ISP Output Window Horizontal Offset Register (0694<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>ISP_OUT_H_OFFS</b>	11:0	rw	<b>Horizontal Picture Offset</b> Unit = pixel
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

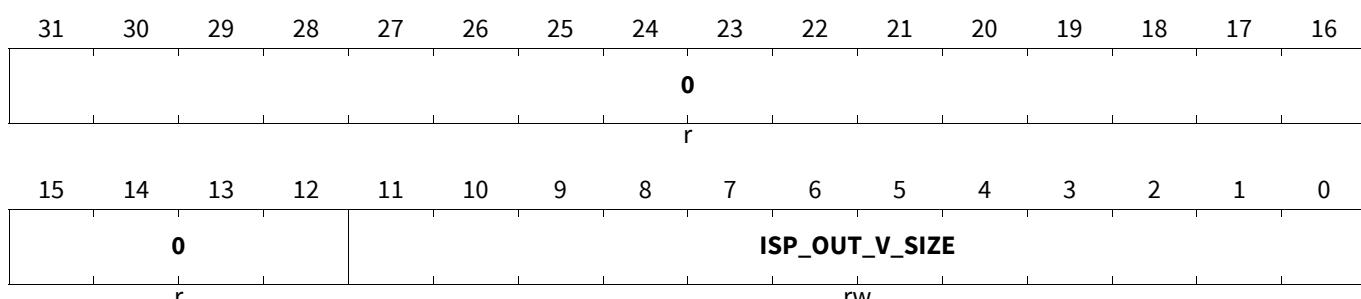
## ISP Output Window Vertical Offset Register

**ISP\_OUT\_V\_OFFSETS****ISP Output Window Vertical Offset Register (0698<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

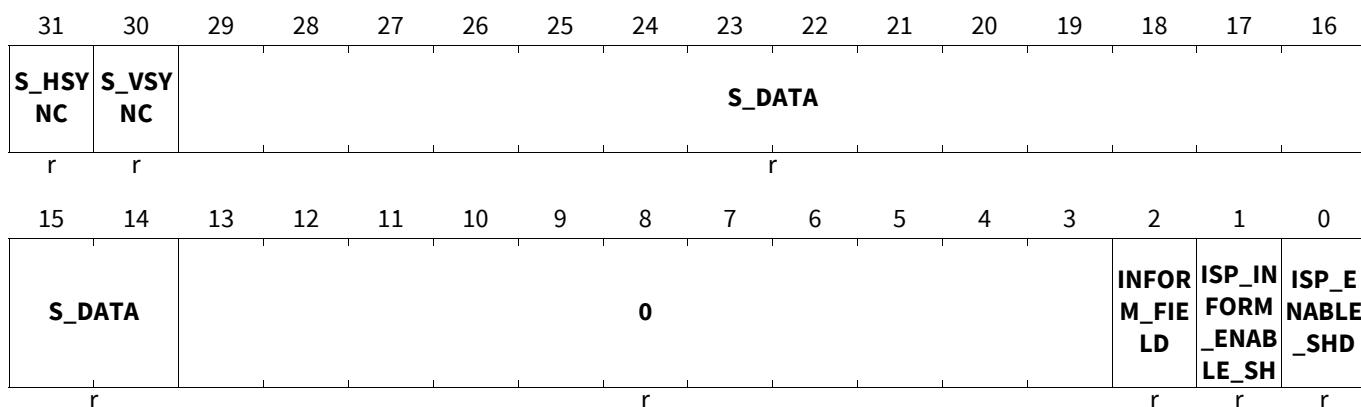
Field	Bits	Type	Description
<b>ISP_OUT_V_OFFSETS</b>	11:0	rw	<b>Vertical Picture Offset</b> Unit = lines
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****ISP Output Horizontal Picture Size Register****ISP\_OUT\_H\_SIZE****ISP Output Horizontal Picture Size Register**(069C<sub>H</sub>)Application Reset Value: 0000 0A28<sub>H</sub>

Field	Bits	Type	Description
ISP_OUT_H_SIZE	11:0	rw	<b>Horizontal picture size</b> in pixel. If ISP_MODE is set to ... <ul style="list-style-type: none"> <li>• 001<sub>B</sub> : ITU-R BT.656 YUV</li> <li>• 010<sub>B</sub> : ITU-R BT.601 YUV</li> </ul> ... only even numbers are accepted, because complete quadruples of YUYV(YCbYCr) are needed for the 422 output. (if an odd size is programmed the value will be truncated to even size)
0	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**ISP Output Vertical Picture Size Register****ISP\_OUT\_V\_SIZE****ISP Output Vertical Picture Size Register**(06A0<sub>H</sub>)Application Reset Value: 0000 0800<sub>H</sub>

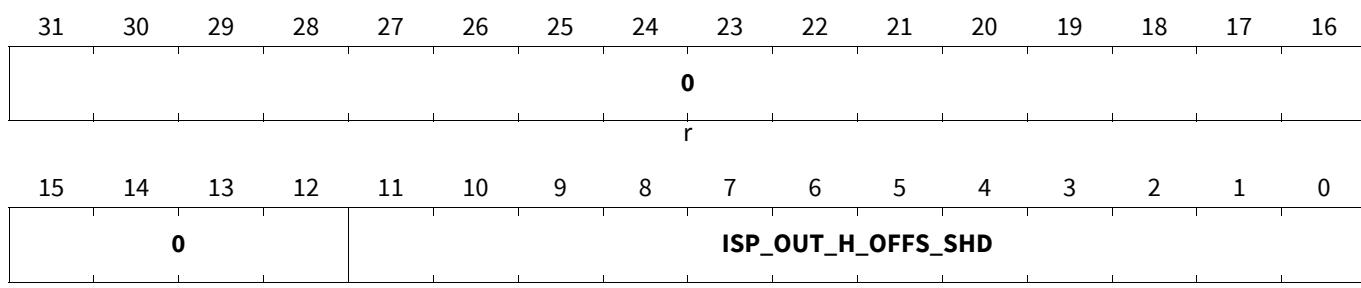
Field	Bits	Type	Description
ISP_OUT_V_SIZE	11:0	rw	<b>Vertical picture size</b> In lines
0	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****ISP Shadow Flags Register****ISP\_FLAGS\_SHD****ISP Shadow Flags Register**(06A8<sub>H</sub>)Application Reset Value: FFFF C000<sub>H</sub>

Field	Bits	Type	Description
<b>ISP_ENABLE_SHD</b>	0	r	<b>ISP enable shadow register</b> 0 <sub>B</sub> no output of data 1 <sub>B</sub> ISP currently outputs data
<b>ISP_INFORM_ENABLE_SHD</b>	1	r	<b>Input formatter enable shadow register</b>
<b>INFORM_FIELD</b>	2	r	<b>Current field information</b> 0 <sub>B</sub> = odd 1 <sub>B</sub> = even
<b>S_DATA</b>	29:14	r	<b>State of ISP input port s_data</b> For test purposes
<b>S_VSYNC</b>	30	r	<b>State of ISP input port s_vsync</b> For test purposes
<b>S_HSYNC</b>	31	r	<b>State of ISP input port s_hsync</b> For test purposes
<b>0</b>	13:3	r	<b>Reserved</b> Read as 0.

**ISP Output Window Horizontal Offset Shadow Register**

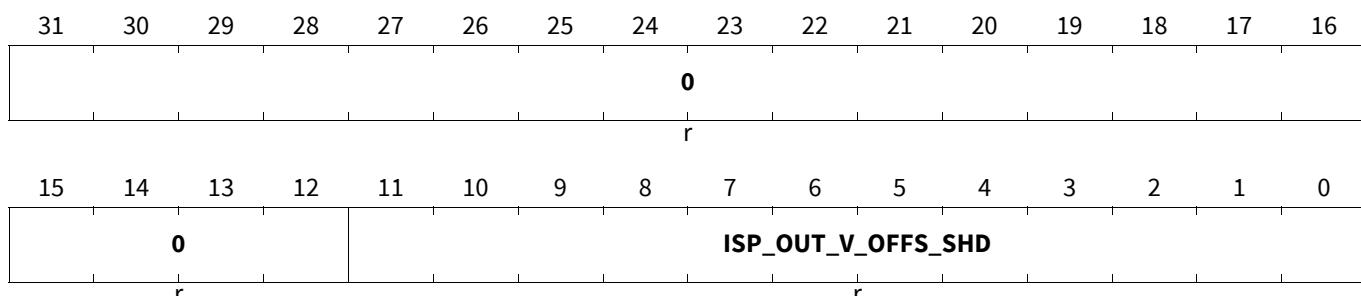
Current horizontal offset of output window (shadow register).

**Camera and ADC Interface (CIF)****ISP\_OUT\_H\_OFFSET\_SHD****ISP Output Window Horizontal Offset Shadow Register(06AC<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>ISP_OUT_H_O FFS_SHD</b>	11:0	r	<b>Current horizontal picture offset</b> In lines
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0.

**ISP Output Window Vertical Offset Shadow Register**

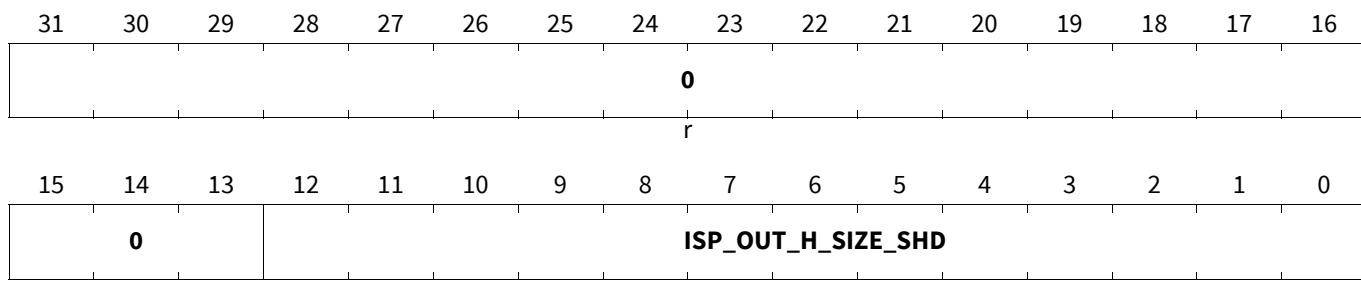
Current vertical offset of output window (shadow register).

**ISP\_OUT\_V\_OFFSET\_SHD****ISP Output Window Vertical Offset Shadow Register(06B0<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>ISP_OUT_V_O FFS_SHD</b>	11:0	r	<b>Current vertical picture offset</b> In lines
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0.

**ISP Output Horizontal Picture Size Shadow Register**

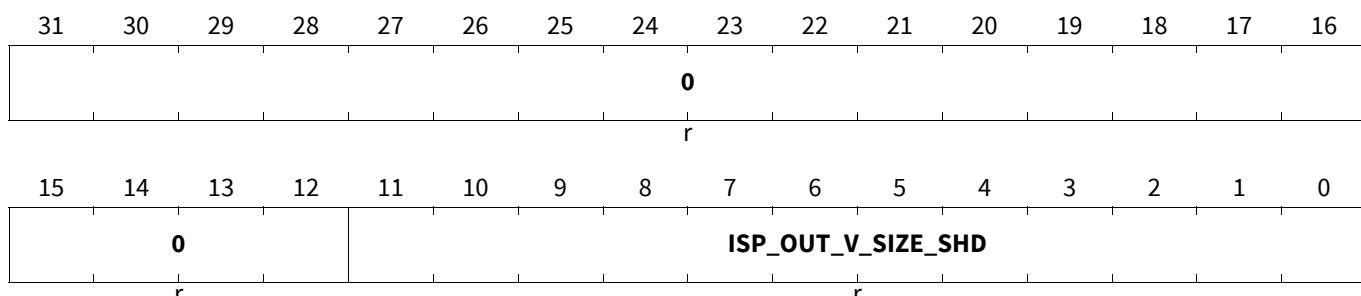
Current output horizontal picture size (shadow register)

**Camera and ADC Interface (CIF)****ISP\_OUT\_H\_SIZE\_SHD****ISP Output Horizontal Picture Size Shadow Register(06B4<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>ISP_OUT_H_SIZE_SHD</b>	12:0	r	<b>Current horizontal picture size</b> In pixels
<b>0</b>	31:13	r	<b>Reserved</b> Read as 0.

**ISP Output Vertical Picture Size Shadow Register**

Current output vertical picture size (shadow register).

**ISP\_OUT\_V\_SIZE\_SHD****ISP Output Vertical Picture Size Shadow Register(06B8<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>ISP_OUT_V_SIZE_SHD</b>	11:0	r	<b>Current vertical pic size</b> In lines
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0.

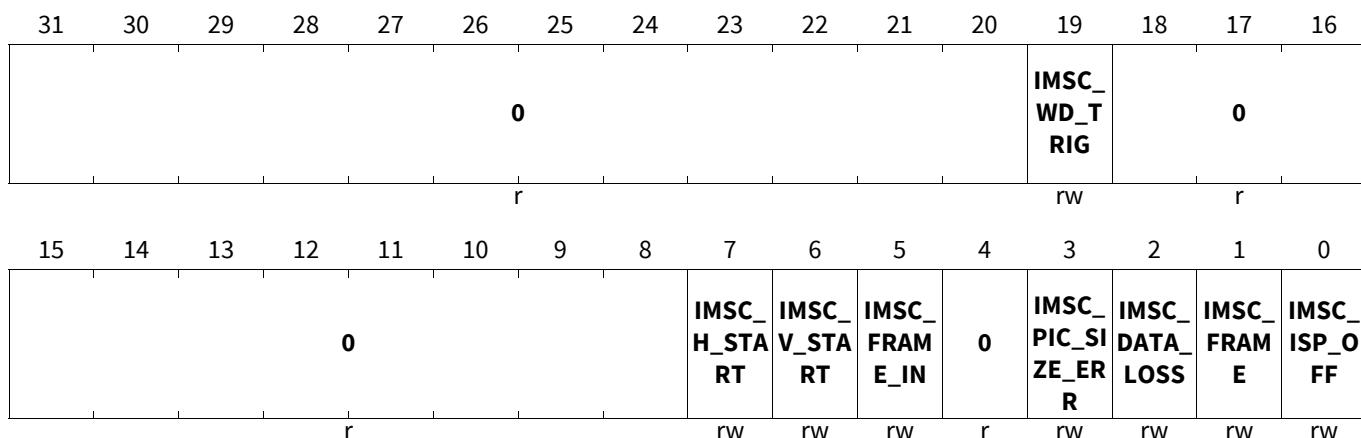
## Camera and ADC Interface (CIF)

## 26.4.2.4 ISP Interrupt Control Registers

## ISP Interrupt Mask Register

## ISP\_IMSC

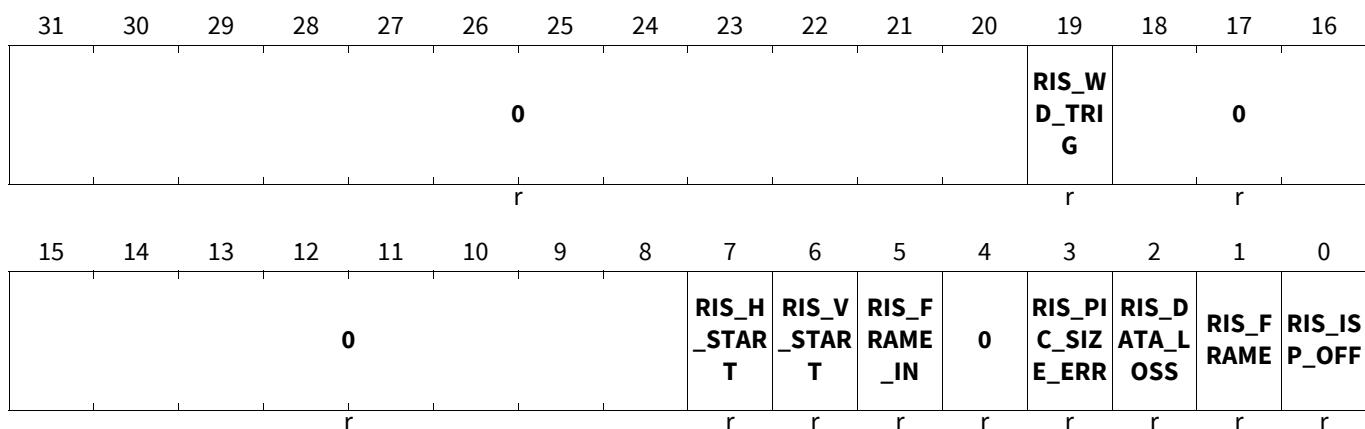
## ISP Interrupt Mask Register

(06BC<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

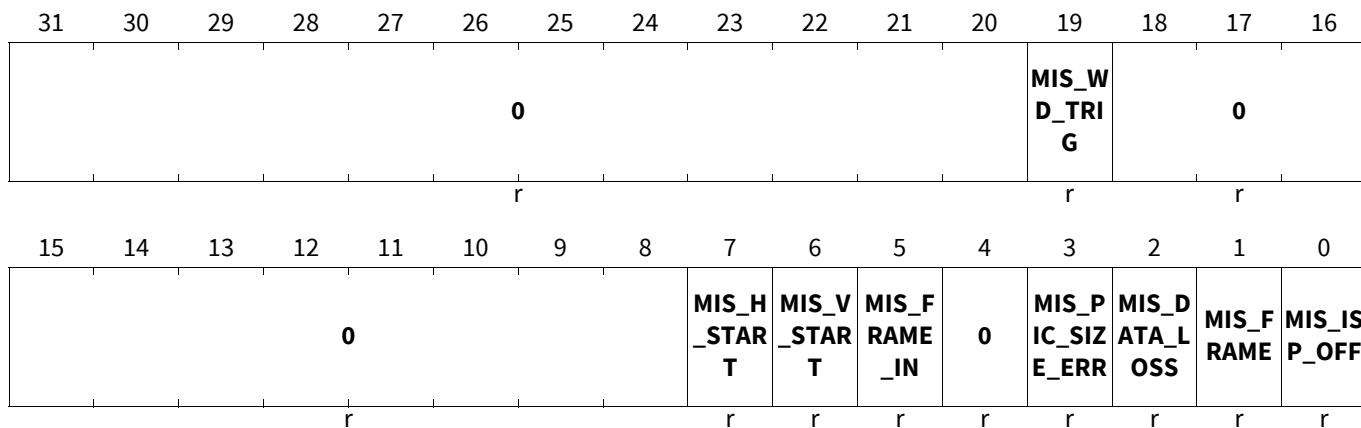
Field	Bits	Type	Description
IMSC_ISP_OF_F	0	rw	<b>Isp was Turned Off (vsynced)</b> Due to f_cnt reached or manual 0 <sub>B</sub> mask out 1 <sub>B</sub> enable interrupt
IMSC_FRAME	1	rw	<b>Frame was Completely Put Out</b> 0 <sub>B</sub> mask out 1 <sub>B</sub> enable interrupt
IMSC_DATA LOSS	2	rw	<b>Loss of Data</b> Within a line, processing failure 0 <sub>B</sub> mask out 1 <sub>B</sub> enable interrupt
IMSC_PIC_SIZE_ERR	3	rw	<b>Pic Size Violation Occurred</b> Programming seems wrong 0 <sub>B</sub> mask out 1 <sub>B</sub> enable interrupt
IMSC_FRAME_IN	5	rw	<b>Sampled Input Frame is Complete</b> 0 <sub>B</sub> mask out 1 <sub>B</sub> enable interrupt
IMSC_V_STAR_T	6	rw	<b>Start Edge of v_sync</b> 0 <sub>B</sub> mask out 1 <sub>B</sub> enable interrupt
IMSC_H_STAR_T	7	rw	<b>Start Edge of h_sync</b> 0 <sub>B</sub> mask out 1 <sub>B</sub> enable interrupt
IMSC_WD_TRI_G	19	rw	<b>A watchdog timeout was triggered at the ISP input</b> 0 <sub>B</sub> mask out 1 <sub>B</sub> enable interrupt

**Camera and ADC Interface (CIF)**

Field	Bits	Type	Description
0	4, 18:8, 31:20	r	<b>Reserved</b> Read as 0, should be written with 0.

**ISP Raw Interrupt Status Register****ISP\_RIS****ISP Raw Interrupt Status Register**(06C0<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>RIS_ISP_OFF</b>	0	r	<b>Isp was Turned Off (vsynced)</b> Due to f_cnt reached or manual
<b>RIS_FRAME</b>	1	r	<b>Frame was Completely Put Out</b>
<b>RIS_DATA_LOSS</b>	2	r	<b>Loss of Data</b> Within a line, processing failure
<b>RIS_PIC_SIZE_ERR</b>	3	r	<b>Pic Size Violation Occurred</b> Programming seems wrong
<b>RIS_FRAME_IN</b>	5	r	<b>Sampled Input Frame is Complete</b>
<b>RIS_V_START</b>	6	r	<b>Start Edge of v_sync</b>
<b>RIS_H_START</b>	7	r	<b>Start Edge of h_sync</b>
<b>RIS_WD_TRIG</b>	19	r	<b>A watchdog timeout was triggered at the ISP input</b>
0	4, 18:8, 31:20	r	<b>Reserved</b>

**Camera and ADC Interface (CIF)****ISP Masked Interrupt Status Register****ISP\_MIS****ISP Masked Interrupt Status Register**(06C4<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

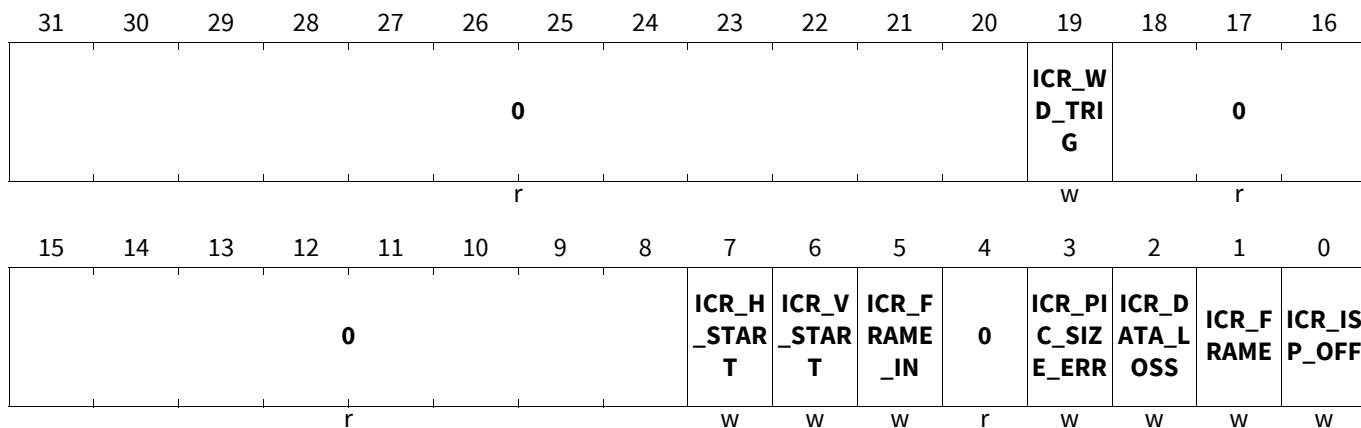
Field	Bits	Type	Description
<b>MIS_ISP_OFF</b>	0	r	<b>Isp was Turned Off (vsynced)</b> Due to f_cnt reached or manual
<b>MIS_FRAME</b>	1	r	<b>Frame was Completely Put Out</b>
<b>MIS_DATA_LOSS</b>	2	r	<b>Loss of Data</b> Within a line, processing failure
<b>MIS_PICTURE_SIZE_ERR</b>	3	r	<b>Pic Size Violation Occurred</b> Programming seems wrong
<b>MIS_FRAME_IN_N</b>	5	r	<b>Sampled Input Frame is Complete</b>
<b>MIS_V_START</b>	6	r	<b>Start Edge of v_sync</b>
<b>MIS_H_START</b>	7	r	<b>Start Edge of h_sync</b>
<b>MIS_WD_TRIG</b>	19	r	<b>A watchdog timeout was triggered at the ISP input</b>
<b>0</b>	4, 18:8, 31:20	r	<b>Reserved</b> Read as 0.

## Camera and ADC Interface (CIF)

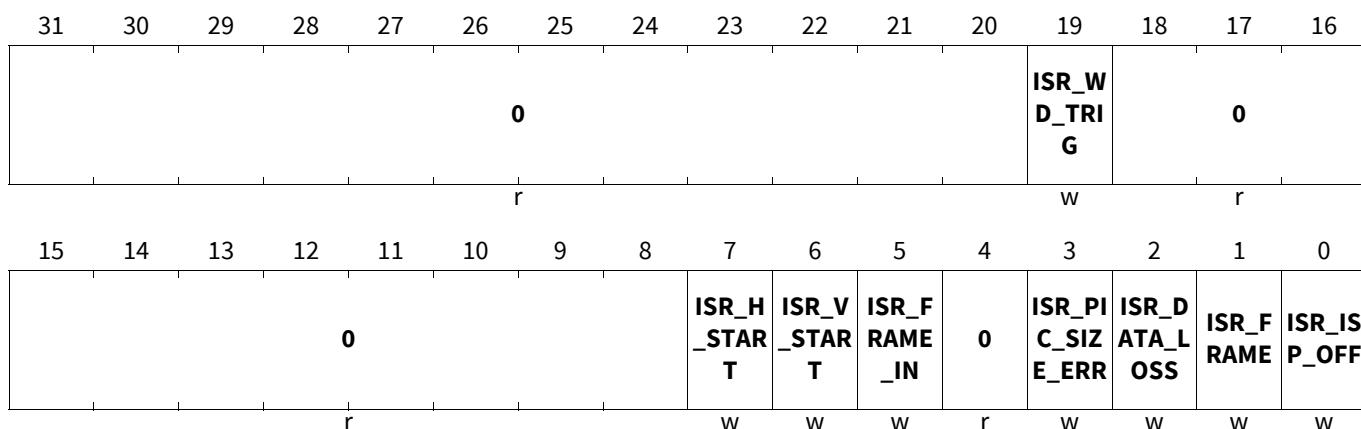
## ISP Interrupt Clear Register

**ISP\_ICR**

## ISP Interrupt Clear Register

(06C8<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>ICR_ISP_OFF</b>	0	w	<b>Isp was Turned Off (vsynced)</b> Clear Interrupt
<b>ICR_FRAME</b>	1	w	<b>Frame was Completely Put Out</b> Clear Interrupt
<b>ICR_DATA_LOSS</b>	2	w	<b>Loss of Data</b> Within a line, processing failure Clear Interrupt
<b>ICR_PIC_SIZE_ERR</b>	3	w	<b>Pic Size Violation Occurred</b> Clear Interrupt
<b>ICR_FRAME_IN</b>	5	w	<b>Sampled Input Frame is Complete</b> Clear Interrupt
<b>ICR_V_START</b>	6	w	<b>Start Edge of v_sync</b> Clear Interrupt
<b>ICR_H_START</b>	7	w	<b>Start Edge of h_sync</b> Clear Interrupt
<b>ICR_WD_TRIG</b>	19	w	<b>A watchdog timeout was triggered at the ISP input</b> Clear Interrupt
<b>0</b>	4, 18:8, 31:20	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****ISP Interrupt Set Register****ISP\_ISR****ISP Interrupt Set Register****(06CC<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>ISR_ISP_OFF</b>	0	w	<b>Isp was Turned Off (vsynced)</b> Set Interrupt
<b>ISR_FRAME</b>	1	w	<b>Frame was Completely Put Out</b> Set Interrupt
<b>ISR_DATA_LOSS</b>	2	w	<b>Loss of Data</b> Within a line, processing failure Set Interrupt
<b>ISR_PIC_SIZE_ERR</b>	3	w	<b>Pic Size Violation Occurred</b> Set Interrupt
<b>ISR_FRAME_IN</b>	5	w	<b>Sampled Input Frame is Complete</b> Set Interrupt
<b>ISR_V_START</b>	6	w	<b>Start Edge of v_sync</b> Set Interrupt
<b>ISR_H_START</b>	7	w	<b>Start Edge of h_sync</b> Set Interrupt
<b>ISR_WD_TRIG</b>	19	w	<b>A watchdog timeout was triggered at the ISP input</b> Set Interrupt
<b>0</b>	4, 18:8, 31:20	r	<b>Reserved</b> Read as 0, should be written with 0.

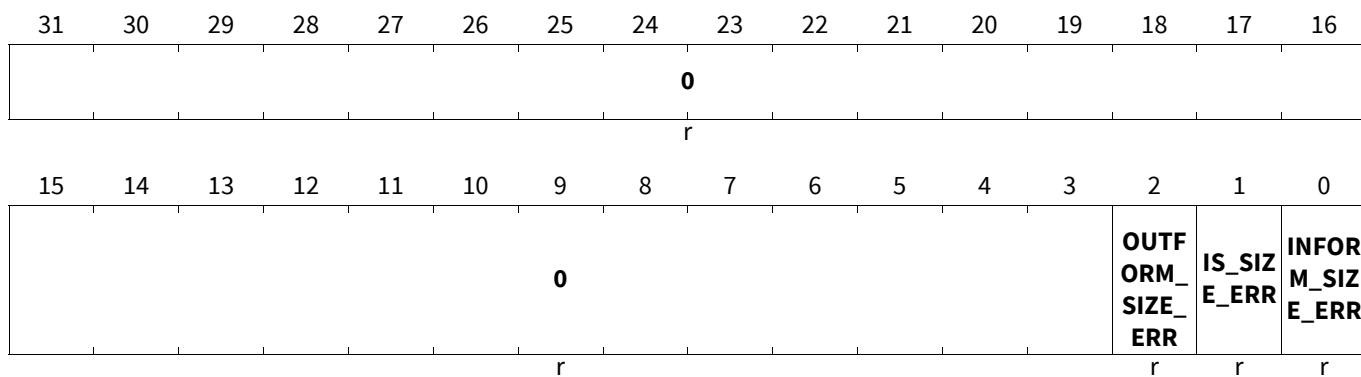
## Camera and ADC Interface (CIF)

## 26.4.2.5 Miscellaneous ISP Registers

## ISP Error Register

## ISP\_ERR

## ISP Error Register

(073C<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
INFORM_SIZE_ERR	0	r	Size error is generated in inform submodule
IS_SIZE_ERR	1	r	Size error is generated in image stabilization submodule
OUTFORM_SIZE_ERR	2	r	Size error is generated in outmux submodule
0	31:3	r	Reserved Read as 0.

## Camera and ADC Interface (CIF)

### ISP\_ERR and ISP\_ERR\_CLR

For debug purposes the **ISP\_ERR** and **ISP\_ERR\_CLR** are implemented. In the case a PIC\_SIZE\_ERR interrupt is signalled the SW is able to see in which submodule this error was generated. Writing to the **ISP\_ERR\_CLR** register clears this bit.

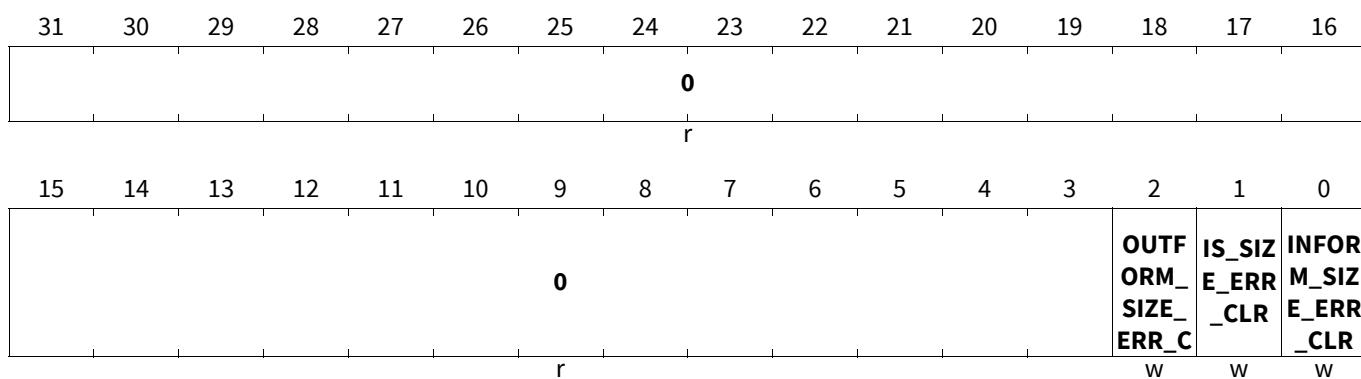
#### ISP Error Clear Register

##### ISP\_ERR\_CLR

##### ISP Error Clear Register

(0740<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
INFORM_SIZE_ERR_CLR	0	w	<b>Size error is cleared</b>
IS_SIZE_ERR_CLR	1	w	<b>Size error is cleared</b>
OUTFORM_SIZE_ERR_CLR	2	w	<b>Size error is cleared</b>
0	31:3	r	<b>Reserved</b> Read as 0, should be written with 0.

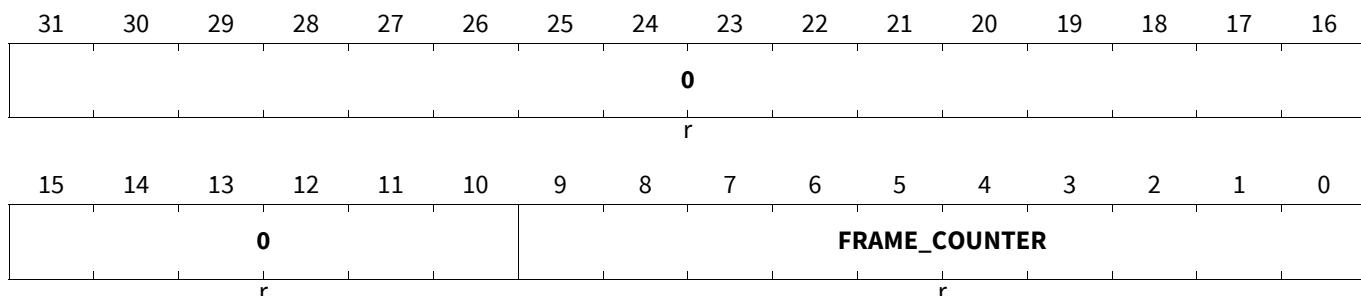
#### ISP Frame Counter Register

##### ISP\_FRAME\_COUNT

##### ISP Frame Counter Register

(0744<sub>H</sub>)

Application Reset Value: 0000 0000<sub>H</sub>



**Camera and ADC Interface (CIF)**

Field	Bits	Type	Description
<b>FRAME_COUN TER</b>	9:0	r	<p><b>Current Frame Count of Processing</b></p> <p>In the ISP_FRAME_COUNT register the number of processed frames are displayed. For example: If a 8 is programmed into the ISP_ACQ_NR_FRAMES register, a read access to the ISP_FRAME_COUNT register during processing of the first picture shows a 7. After the entire frames are processed the ISP_OFF interrupt is generated and the ISP_FRAME_COUNT has the count zero. In case a '0' is programmed into the ISP_ACQ_NR_FRAMES register (continues mode) the ISP_FRAME_COUNT register keeps the value '0'.</p>
<b>0</b>	31:10	r	<p><b>Reserved</b></p> <p>Read as 0.</p>

## Camera and ADC Interface (CIF)

## 26.4.3 Linear Downscaler Programming Registers

The address of each CIF Linear Downscaler register is evaluated as CIF\_LIN\_DSSCALE\_BASE + Offset.

## 26.4.3.1 Linear Downscaler Configuration Registers

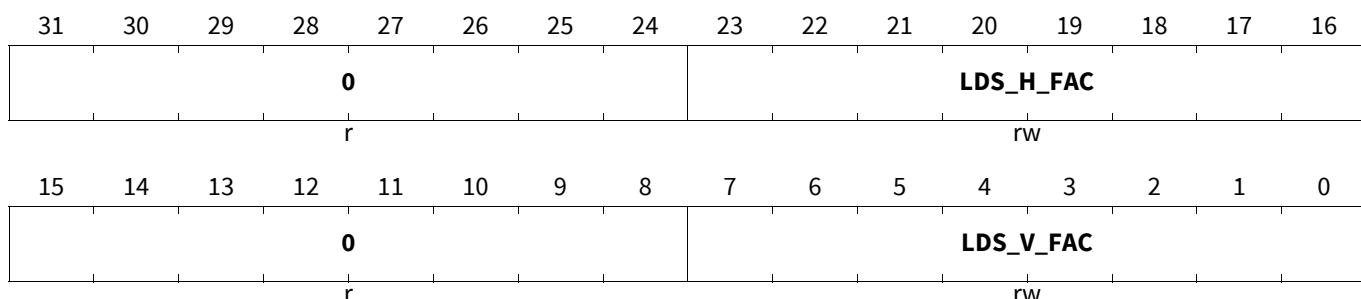
## Linear Downscaler Control Register

## LDS\_CTRL

Linear Downscaler Control Register (2600<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					LDS_H_MODE	0	LDS_V_MODE	0	LDS_H_EN	LDS_V_EN					

Field	Bits	Type	Description
LDS_V_EN	0	rw	<b>Vertical scaling enable</b> 0 <sub>B</sub> Vertical downscaling is disabled 1 <sub>B</sub> Vertical downscaling is enabled
LDS_H_EN	1	rw	<b>Horizontal scaling enable</b> 0 <sub>B</sub> Horizontal downscaling is disabled 1 <sub>B</sub> Horizontal downscaling is enabled
LDS_V_MODE	5:4	rw	<b>Vertical scaling mode</b> 00 <sub>B</sub> Single skip 01 <sub>B</sub> Double skip 10 <sub>B</sub> Single pass 11 <sub>B</sub> Double pass
LDS_H_MODE	9:8	rw	<b>Horizontal scaling mode</b> 00 <sub>B</sub> Single skip 01 <sub>B</sub> Double skip 10 <sub>B</sub> Single pass 11 <sub>B</sub> Double pass
0	3:2, 7:6, 31:10	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****Linear Downscaler Factor Register****LDS\_FAC****Linear Downscaler Factor Register**(2604<sub>H</sub>)**Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>LDS_V_FAC</b>	7:0	rw	<b>Vertical scaling factor</b> Depending on the configured mode every lds_v_fac+1th line (or double line) will be skipped or passed on to the next module.
<b>LDS_H_FAC</b>	23:16	rw	<b>Horizontal scaling factor</b> Depending on the configured mode every lds_h_fac+1th pixel (or double pixel) will be skipped or passed on to the next module.
<b>0</b>	15:8, 31:24	r	<b>Reserved</b> Read as 0, should be written with 0.

## Camera and ADC Interface (CIF)

### 26.4.4 Memory Interface Programming Registers

The address of each CIF memory interface register is evaluated as CIF\_MI\_BASE + Offset.

#### 26.4.4.1 Memory Interface Control Registers

##### Memory Interface Global Control Register

Note: *In order to select the appropriate mode, set one bit from RAW\_ENABLE, JPEG\_ENABLE and MP\_ENABLE to 1, and set the others to 0.*

##### MI\_CTRL

**Memory Interface Global Control Register (1500<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								MP_WRITE_FORMAT	INIT_OFFSET_EN	INIT_BASE_N	0	BURST_LEN_CHROM	0	BURST_LEN_LUM	
				0					r	rw	rw	r	rw	r	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								BYTE_SWAP		0		RAW_ENABLE	JPEG_ENABLE	0	MP_ENABLE
				0					r	rw	r	rw	rw	r	rw

Field	Bits	Type	Description
<b>MP_ENABLE</b>	0	rw	<b>Enables main picture data path, YCbCr mode</b> Programmed value becomes effective (visible in control shadow register) after a soft reset, a forced software update or an automatic config update. Affects MI_IN and MI_OUT module.
<b>JPEG_ENABLE</b>	2	rw	<b>Enables JPEG mode</b> Programmed value becomes effective (visible in control shadow register) after a soft reset or a forced software. Affects MI_IN and MI_OUT module.
<b>RAW_ENABLE</b>	3	rw	<b>Enables RAW mode</b> Programmed value becomes effective (visible in control shadow register) after a soft reset or a forced software update. Affects MI_IN and MI_OUT module.
<b>BYTE_SWAP</b>	7	rw	<b>Byte Swap Enable</b> Enables change of byte order of the 32 bit output word at write port.  Note: <i>Programmed value becomes effective immediately. So write to the register only if no picture data is sent.</i>  0 <sub>B</sub> no byte mirroring 1 <sub>B</sub> byte order is mirrored but the bit order within one byte doesn't change

## Camera and ADC Interface (CIF)

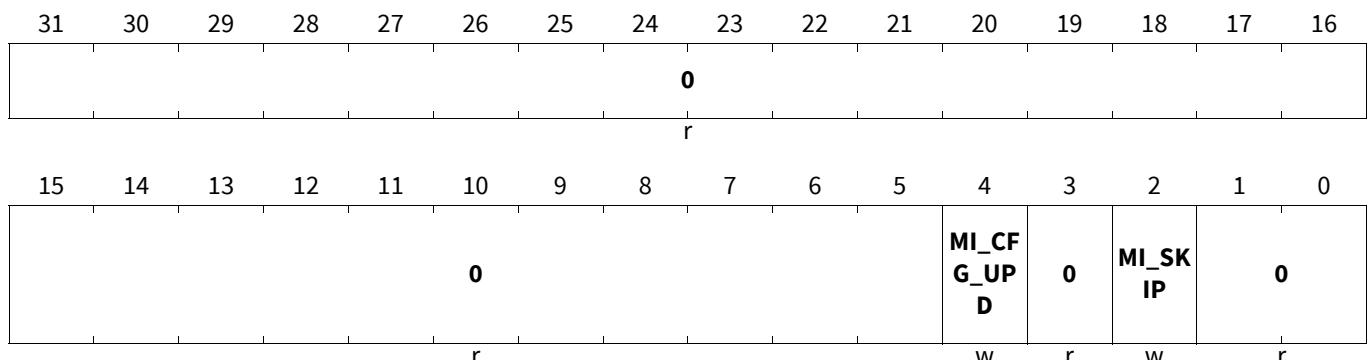
Field	Bits	Type	Description
<b>BURST_LEN_LUM</b>	16	rw	<p><b>Luminance Burst Length</b>            Burst length for Y, JPEG, or RAW data affecting write port.            Ignored if 8-beat bursts not supported.            This burst configuration is also applied to the Extra Paths.</p> <p><i>Note:</i> Programmed value becomes effective immediately. So write to the register only if no picture data is sent.</p> <p>0<sub>B</sub> 4-beat bursts            1<sub>B</sub> 8-beat bursts</p>
<b>BURST_LEN_CHROM</b>	18	rw	<p><b>Chrominance Burst Length</b>            Burst length for Cb or Cr data affecting write port.            Ignored if 8-beat bursts not supported.</p> <p><i>Note:</i> Programmed value becomes effective immediately. So write to the register only if no picture data is sent.</p> <p>0<sub>B</sub> 4-beat bursts            1<sub>B</sub> 8-beat bursts</p>
<b>INIT_BASE_EN</b>	20	rw	<p><b>Init Base Address Enable</b>            Enables updating of the base address and buffer size shadow registers for picture to the programmed register init values.</p> <p>MI_MP_Y/CB/CR_BASE_AD_INIT            -&gt; MI_MP_Y/CB/CR_BASE_AD_SHD            MI_MP_Y/CB/CR_SIZE_INIT            -&gt; MI_MP_Y/CB/CR_SIZE_SHD</p> <p>The update will be executed either when a forced software update occurs (in register MI_INIT bit cfg_upd = 1) or when an automatic config update signal arrives at the MI input port. So only the corresponding shadow registers are affected.</p>
<b>INIT_OFFSET_EN</b>	21	rw	<p><b>Init Offset Counter Enable</b>            Enables updating of the offset counters shadow registers for picture to the programmed register init values.</p> <p>MI_MP_Y/CB/CR_OFFSET_CNT_INIT            -&gt; MI_MP_Y/CB/CR_OFFSET_CNT_SHD</p> <p>The update will be executed either when a forced software update occurs (in register MI_INIT bit cfg_upd = 1) or when an automatic config update signal arrives at the MI input port. So only the corresponding shadow registers are affected.</p> <p>After a picture skip has been performed init_offset_en selects between skip restart and skip init mode (see bit skip in register MI_INIT).</p>

**Camera and ADC Interface (CIF)**

Field	Bits	Type	Description
<b>MP_WRITE_FORMAT</b>	23:22	rw	<p><b>Main Picture YCbCr Write Format</b>  Defines how YCbCr main picture data is written to memory.  Ignored if JPEG data is chosen.</p> <p>The description listed first in the table below indicates the write format as used in YCbCr mode, the second one is valid for RAW data mode only.</p> <p><i>Note:</i> Programmed value becomes effective immediately. So write to the register only if no picture data is sent to the path.</p> <ul style="list-style-type: none"> <li>00<sub>B</sub> planar (YCbCr mode) / RAW &amp; data mode (8 bit)</li> <li>01<sub>B</sub> semi-planar for YCbCr 4:2:x / RAW 8 bit</li> <li>10<sub>B</sub> interleaved_combined for YCbCr 4:2:2 only / RAW &amp; data mode (greater 8 up to 16 bit)</li> <li>11<sub>B</sub> reserved / RAW greater 8 up to 16 bit</li> </ul>
<b>0</b>	1, 6:4, 15:8, 17, 19, 31:24	r	<p><b>Reserved</b>  Read as 0, should be written with 0.</p>

**Memory Interface Control Register For Address Init And Skip Function Register****MI\_INIT**

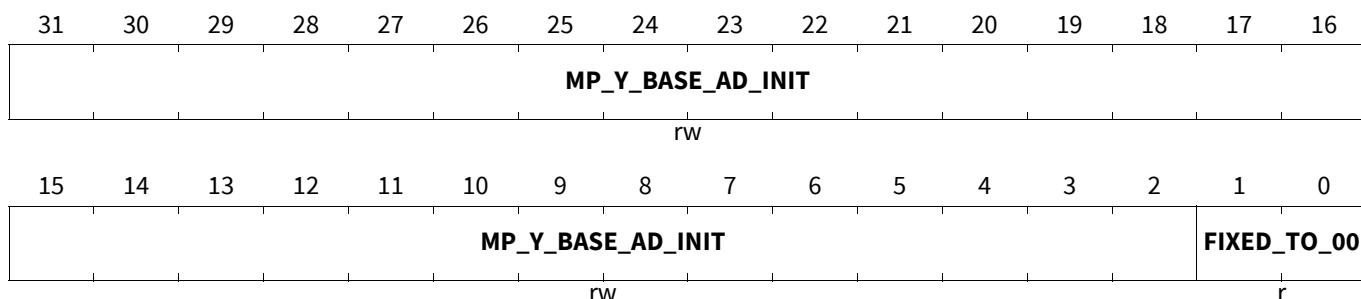
**Memory Interface Control Register For Address Init And Skip Function Register(1504<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**



## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
<b>MI_SKIP</b>	2	w	<p><b>Skip Picture</b></p> <p>Skip of current or next starting main picture: Aborts writing of main picture image data of the current frame to RAM (after the current burst transmission has been completed). Further main picture data up to the end of the current frame are discarded.No further macroblock line interrupt (_line), no wrap around interrupt for main picture (wrap_mp_y/cb/cr) and no fill level interrupt (fill_mp_y) are generated.Skip does not affect the generation of the main path frame end interrupt (mp_frame_end).The byte counter (register MI_BYTE_CNT) is not affected. It produces the correct number of JPEG or RAW data bytes at the end of the current (skipped) frame.After a skip has been performed the offset counter for the main picture at the start of the following frame are set depending on the bit init_offset_en in register MI_CTRL:</p> <p>A) Skip restart mode (init_offset_en = 0) The offset counters of the main picture are restarted at the old start value of the previous skipped frame.</p> <p>B) Skip init mode (init_offset_en = 1) The offset counters of the main picture are initialized with the register contents of the offset counter init registers without any additional forced software update or automatic config update.</p>
<b>MI_CFG_UPD</b>	4	w	<p><b>Forced Configuration Update</b></p> <p>Leads to an immediate update of the shadow registers. Depending on the two init enable bits in the MI_CTRL register (init_offset_en and init_base_en) the offset counter, base address and buffer size shadow registers are also updated.</p>
0	1:0, 3, 31:5	r	<p><b>Reserved</b></p> <p>Read as 0, should be written with 0.</p>

## Memory Interface Base Address For Main Picture Y Component, JPEG or RAW Data Register

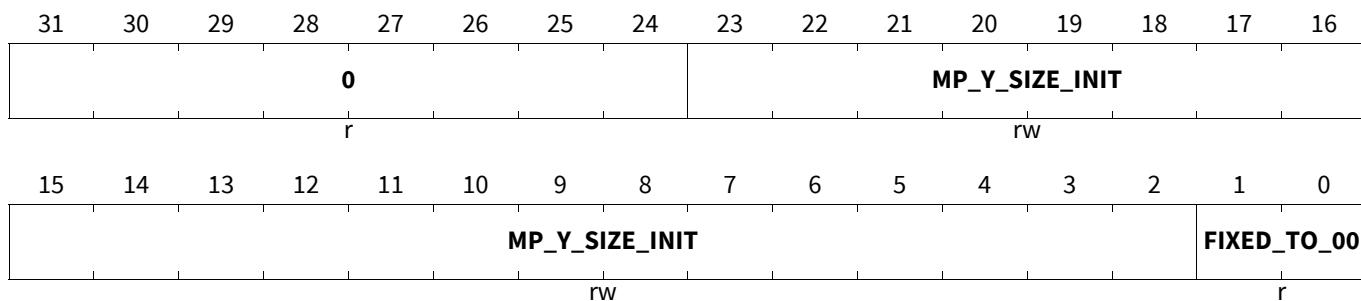
**MI\_MP\_Y\_BASE\_AD\_INIT**Memory Interface Base Address For Main Picture Y Component, JPEG or RAW Data Register(1508<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_Y_BASE_A</b>	31:2	rw	<p><b>Main Picture Y Base Address Init</b></p> <p>Base address of main picture Y component ring buffer, JPEG ring buffer or RAW data ring buffer.</p> <p>Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.</p> <p><i>Note:</i> Set control bit <i>init_base_en</i> before updating so that a forced or automatic update can take effect.</p>

**Memory Interface Size of main picture Y component, JPEG or RAW data Register****MI\_MP\_Y\_SIZE\_INIT**

**Memory Interface Size of main picture Y component, JPEG or RAW data Register(150C<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_Y_SIZE_INIT</b>	23:2	rw	<p><b>Main Picture Y Size Init</b></p> <p>Size of main picture Y component ring buffer, JPEG ring buffer or RAW data ring buffer.</p> <p>Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.</p> <p><i>Note:</i> Set control bit <i>init_base_en</i> before updating so that a forced or automatic update can take effect.</p>
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****Memory Interface Offset Counter Init Value For Main Picture Y, JPEG or RAW Data Register****MI\_MP\_Y\_OFFSET\_CNT\_INIT****Memory Interface Offset Counter Init Value For Main Picture Y, JPEG or RAW Data Register(1510<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								MP_Y_OFFSET_CNT_INIT							
r															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MP_Y_OFFSET_CNT_INIT								FIXED_TO_00							
															r

Field	Bits	Type	Description
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value).
MP_Y_OFFSET_CNT_INIT	23:2	rw	<p><b>Main Picture Y Offset Counter Init</b>  Offset counter init value of main picture Y component ring buffer, JPEG ring buffer or RAW data ring buffer.  Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.</p> <p><i>Note:</i> Set control bit <i>init_base_en</i> before updating so that a forced or automatic update can take effect. Check exceptional handling in skip modes.</p>
0	31:24	r	<p><b>Reserved</b>  Read as 0, should be written with 0.</p>

**Memory Interface Offset Counter Start Value For Main Picture Y, JPEG or RAW Data Register****MI\_MP\_Y\_OFFSET\_CNT\_START****Memory Interface Offset Counter Start Value For Main Picture Y, JPEG or RAW Data Register(1514<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								MP_Y_OFFSET_CNT_START							
r															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MP_Y_OFFSET_CNT_START								FIXED_TO_00							
															r

Field	Bits	Type	Description
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value).

## Camera and ADC Interface (CIF)

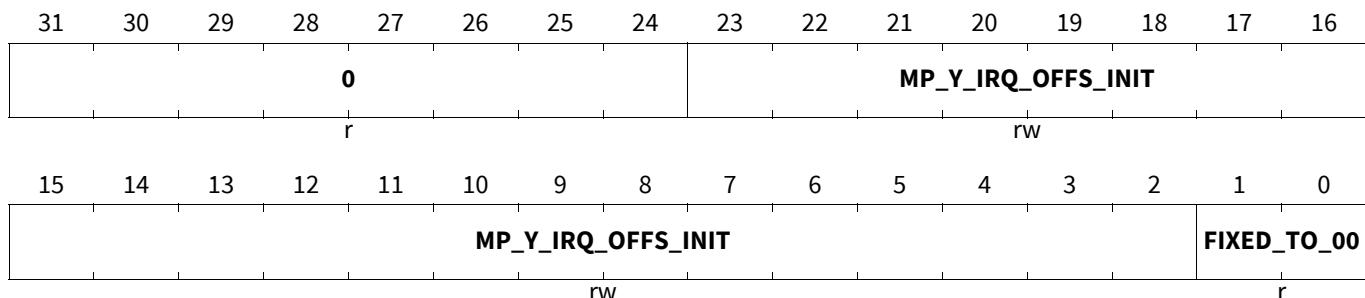
Field	Bits	Type	Description
<b>MP_Y_OFFSET_C</b>	23:2	r	<b>Main Picture Y Offset Counter Start</b> Offset counter value which points to the start address of the previously processed picture (main picture Y component, JPEG or RAW data). Updated at frame end. Note: A soft reset resets the contents to the reset value.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

**Memory Interface Fill Level Interrupt Offset Value For Main Picture Y, JPEG or RAW Data Register**

Register 236

**MI\_MP\_Y\_IRQ\_OFFSET\_INIT****Memory Interface Fill Level Interrupt Offset Value For Main Picture Y, JPEG or RAW Data Register(1518H)**

Application Reset Value: 0000 0000H



Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_Y_IRQ_OF</b>	23:2	rw	<b>Main Picture Y IRQ Offset Init</b> Reaching this programmed value by the current offset counter for addressing main picture Y component, JPEG or RAW data leads to generation of fill level interrupt fill_mp_y. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****Memory Interface Base Address For Main Picture Cb Component Ring Buffer Register****MI\_MP\_CB\_BASE\_AD\_INIT****Memory Interface Base Address For Main Picture Cb Component Ring Buffer Register(151C<sub>H</sub>) Application****Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>MP_CB_BASEAD_INIT</b>															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>MP_CB_BASEAD_INIT</b>															<b>FIXED_TO_00</b>
rw															

Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_CB_BASE AD_INIT</b>	31:2	rw	<b>Main Picture Cb Base Address Init</b> Base address of main picture Cb component ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.  <i>Note:</i> Set control bit init_base_en before updating so that a forced or automatic update can take effect.

**Memory Interface Size Of Main Picture Cb Component Ring Buffer Register**

Register 238

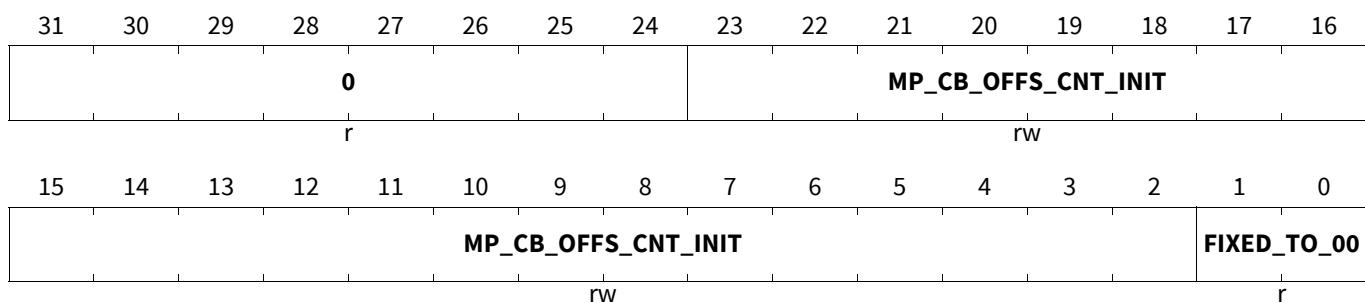
**MI\_MP\_CB\_SIZE\_INIT****Memory Interface Size Of Main Picture Cb Component Ring Buffer Register(1520<sub>H</sub>) Application Reset****Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
<b>0</b>															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>MP_CB_SIZE_INIT</b>															<b>FIXED_TO_00</b>
rw															

Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>

**Camera and ADC Interface (CIF)**

Field	Bits	Type	Description
<b>MP_CB_SIZE_INIT</b>	23:2	rw	<p><b>Main Picture Cb Size Init</b>            Size of main picture Cb component ring buffer.            Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.</p> <p><i>Note:</i> Set control bit init_base_en before updating so that a forced or automatic update can take effect.</p>
<b>0</b>	31:24	r	<p><b>Reserved</b>            Read as 0, should be written with 0.</p>

**Memory Interface Offset Counter Init Value For Main Picture Cb Component Ring Buffer Register****MI\_MP\_CB\_OFFSET\_CNT\_INIT****Memory Interface Offset Counter Init Value For Main Picture Cb Component Ring Buffer Register(1524H)****Application Reset Value: 0000 0000H**

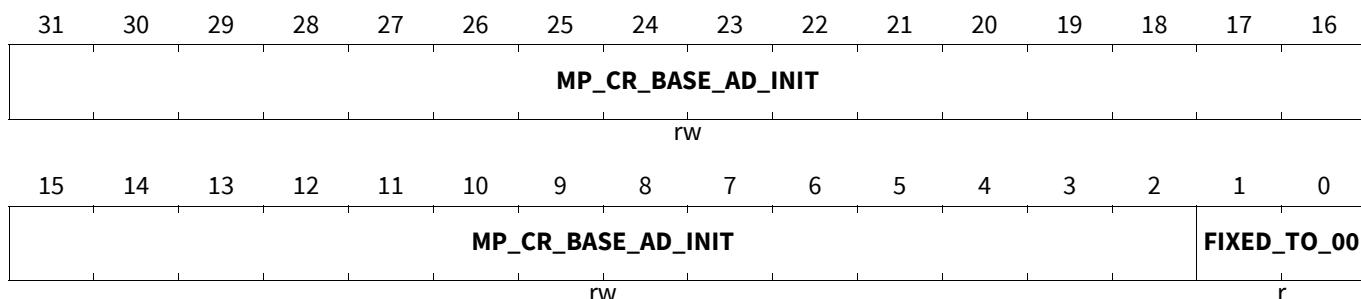
Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_CB_OFFSET_CNT_INIT</b>	23:2	rw	<p><b>Main Picture Cb Offset Counter Init</b>            Offset counter init value of main picture Cb component ring buffer.            Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.</p> <p><i>Note:</i> Set control bit init_base_en before updating so that a forced or automatic update can take effect. Check exceptional handling in skip modes.</p>
<b>0</b>	31:24	r	<p><b>Reserved</b>            Read as 0, should be written with 0.</p>

**Memory Interface Offset Counter Start Value For Main Picture Cb Component Ring Buffer Register**

Register 240

**Camera and ADC Interface (CIF)****MI\_MP\_CB\_OFFSET\_CNT\_START****Memory Interface Offset Counter Start Value For Main Picture Cb Component Ring Buffer Register(1528H)****Application Reset Value: 0000 0000H**

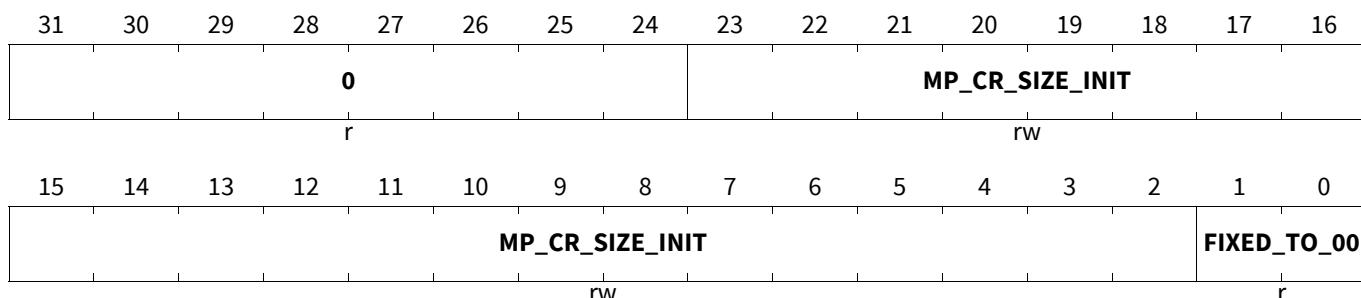
Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_CB_OFFSET_CNT_START</b>	23:2	r	<b>Main Picture Cb Offset Count Start</b> Offset counter value which points to the start address of the previously processed picture (main picture Cb component). Updated at frame end.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

**Memory Interface Base Address For Main Picture Cr Component Ring Buffer Register****MI\_MP\_CR\_BASE\_AD\_INIT****Memory Interface Base Address For Main Picture Cr Component Ring Buffer Register(152CH)** Application Reset Value: 0000 0000H

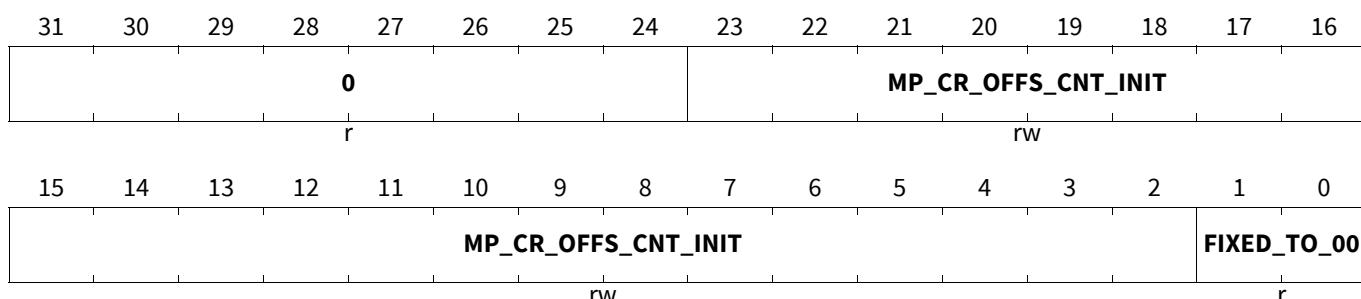
Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_CR_BASE_AD_INIT</b>	31:2	rw	<b>Main Picture Cr Base Address Init</b> Base address of main picture Cr component ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.  Note: Set control bit <i>init_base_en</i> before updating so that a forced or automatic update can take effect.

**Camera and ADC Interface (CIF)****Memory Interface Size Of Main Picture Cr Component Ring Buffer Register**

Register 242

**MI\_MP\_CR\_SIZE\_INIT**
**Memory Interface Size Of Main Picture Cr Component Ring Buffer Register(1530<sub>H</sub>)** Application Reset Value:  
**0000 0000<sub>H</sub>**


Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_CR_SIZE_INIT</b>	23:2	rw	<b>Main Picture Cr Size Init</b> Size of main picture Cr component ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.  <i>Note:</i> Set control bit init_base_en before updating so that a forced or automatic update can take effect.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0, should be written with 0.

**Memory Interface Offset Counter Init value For Main Picture Cr Component Ring Buffer Register****MI\_MP\_CR\_OFFS\_CNT\_INIT**
**Memory Interface Offset Counter Init value For Main Picture Cr Component Ring Buffer Register(1534<sub>H</sub>)** Application Reset Value: **0000 0000<sub>H</sub>**


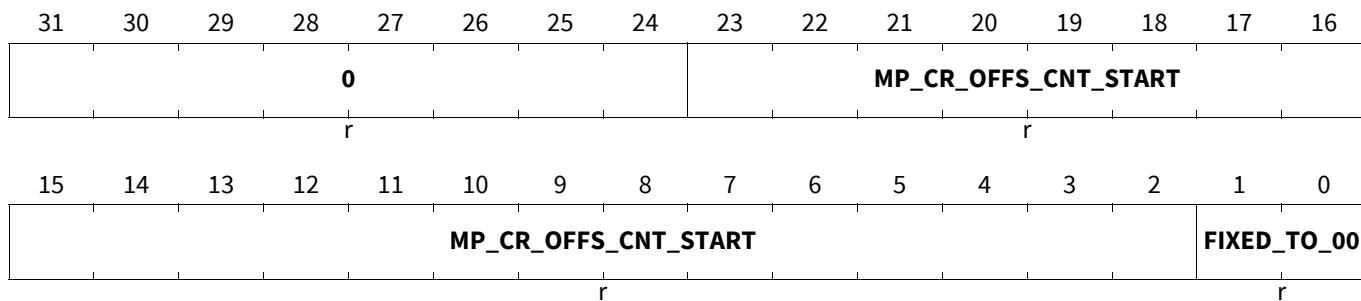
Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>

**Camera and ADC Interface (CIF)**

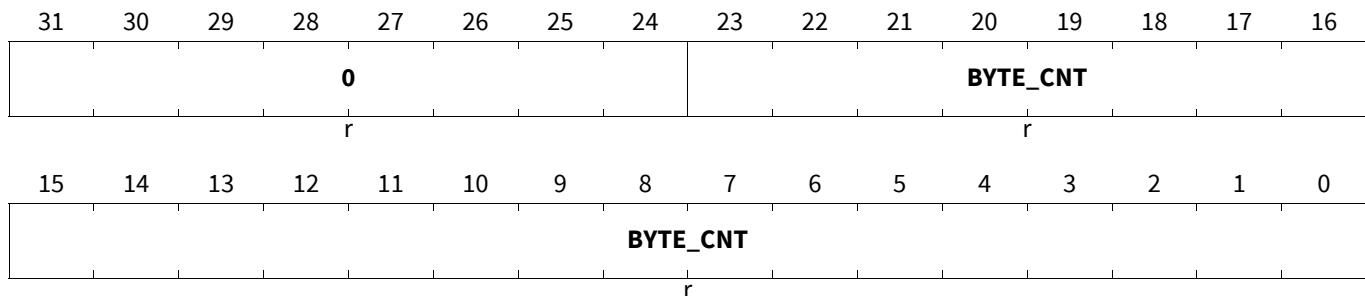
Field	Bits	Type	Description
<b>MP_CR_OFFSET_CNT_INIT</b>	23:2	rw	<p><b>Main Picture Cr Offset Counter Init</b></p> <p>Offset counter init value of main picture Cr component ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.</p> <p><i>Note:</i> Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect. Check exceptional handling in skip modes.</p>
<b>0</b>	31:24	r	<p><b>Reserved</b></p> <p>Read as 0, should be written with 0.</p>

**Memory Interface Offset Counter Start Value For Main Picture Cr Component Ring Buffer Register****MI\_MP\_CR\_OFFSET\_CNT\_START**

**Memory Interface Offset Counter Start Value For Main Picture Cr Component Ring Buffer Register(1538<sub>H</sub>)**  
**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_CR_OFFSET_CNT_START</b>	23:2	r	<p><b>Main Picture Cr Offset Counter Start</b></p> <p>Offset counter value which points to the start address of the previously processed picture (main picture Cr component). Updated at frame end.</p> <p><i>Note:</i> Soft reset will reset the contents to reset value.</p>
<b>0</b>	31:24	r	<p><b>Reserved</b></p> <p>Read as 0.</p>

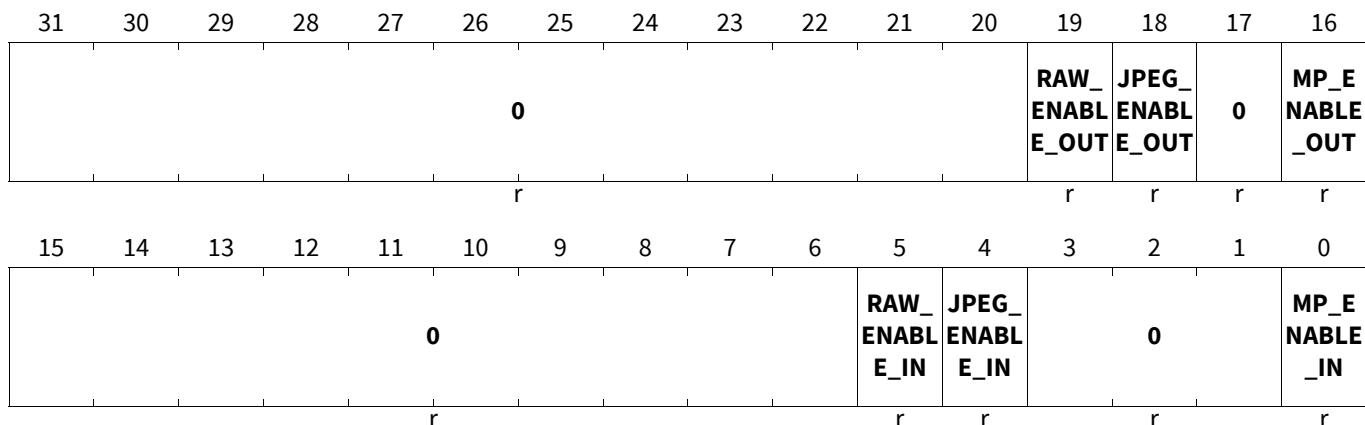
**Camera and ADC Interface (CIF)****Memory Interface Counter Value of JPEG or RAW Data Bytes Register****MI\_BYTE\_CNT****Memory Interface Counter Value of JPEG or RAW Data Bytes Register(1570<sub>H</sub>) Application Reset Value: 0000****0000<sub>H</sub>**

Field	Bits	Type	Description
<b>BYTE_CNT</b>	23:0	r	<b>Byte Count</b> Counter value specifies the number of JPEG or RAW data bytes of the last transmitted frame. Updated at frame end. A soft reset will set the byte counter to zero.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

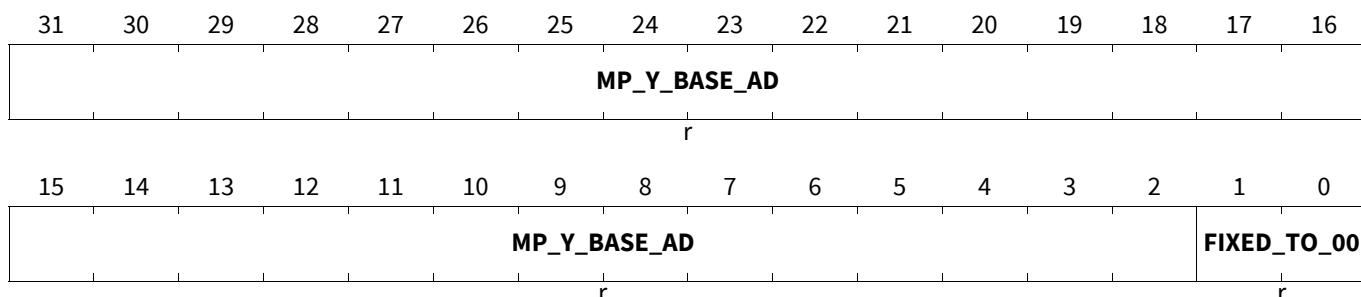
## Camera and ADC Interface (CIF)

## 26.4.4.2 Memory Interface Shadow Registers

## Memory Interface Global Control Internal Shadow Register

**MI\_CTRL\_SHD**Memory Interface Global Control Internal Shadow Register(1574<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>

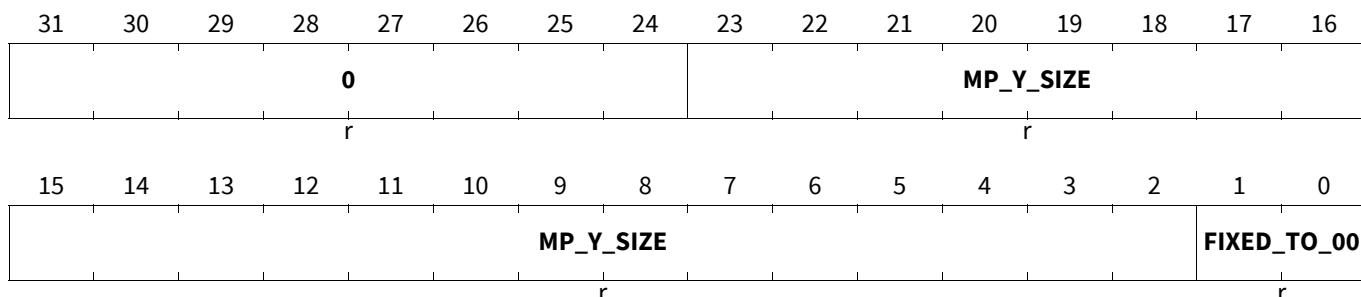
Field	Bits	Type	Description
MP_ENABLE_IN	0	r	<b>Main Picture In Enable</b> Main picture data used in module MI_IN
JPEG_ENABLE_IN	4	r	<b>JPEG In Enable</b> JPEG data used in module MI_IN
RAW_ENABLE_IN	5	r	<b>RAW In Enable</b> RAW data used in module MI_IN
MP_ENABLE_OUT	16	r	<b>Main Picture Out Enable</b> Main picture used in module MI_OUT
JPEG_ENABLE_OUT	18	r	<b>JPEG Out Enable</b> JPEG data used in module MI_OUT
RAW_ENABLE_OUT	19	r	<b>RAW Out Enable</b> Raw data used in module MI_OUT
0	3:1, 15:6, 17, 31:20	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****Memory Interface Base Address Shadow Register For Main Picture Y Component, JPEG Register****MI\_MP\_Y\_BASE\_AD\_SHD****Memory Interface Base Address Shadow Register For Main Picture Y Component, JPEG Register(1578<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

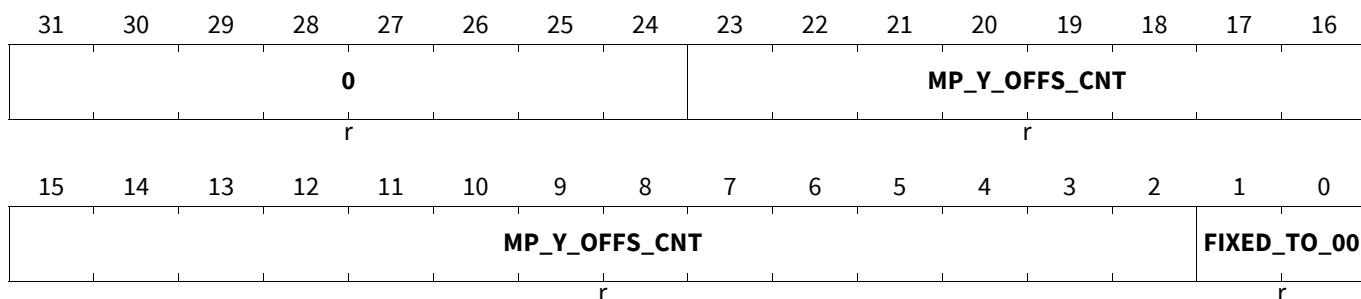
Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	Bits [1:0] are set to “00” (word aligned value).
<b>MP_Y_BASE_A</b>	31:2	r	<b>Main Picture Y Base Address</b> Base address of main picture Y component ring buffer, JPEG ring buffer or RAW data ring buffer

**Memory Interface Size Shadow Register of Main Picture Y Component,JPEG or RAW Data Register**

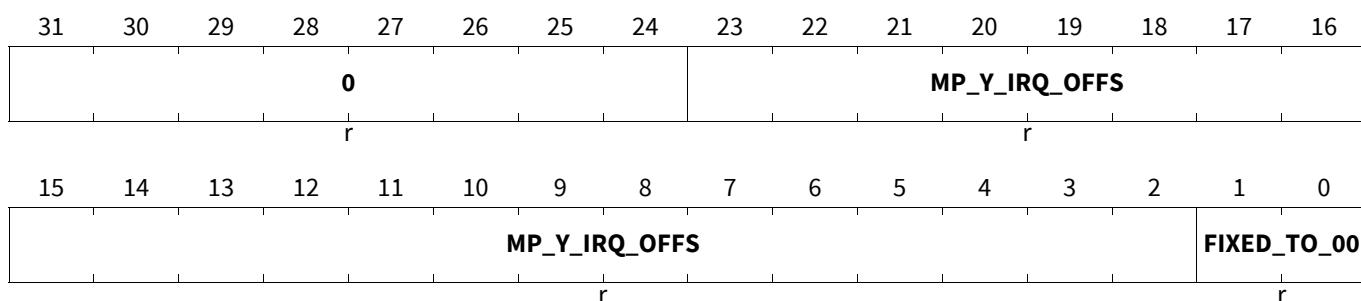
Register 264

**MI\_MP\_Y\_SIZE\_SHD****Memory Interface Size Shadow Register of Main Picture Y Component,JPEG or RAW Data Register(157C<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

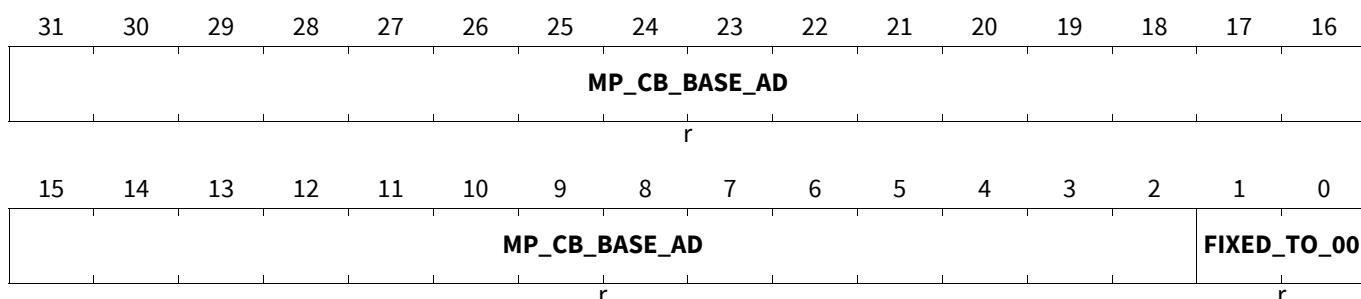
Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	Bits [1:0] are set to “00” (word aligned value).
<b>MP_Y_SIZE</b>	23:2	r	<b>Main Picture Y Size</b> Size of main picture Y component ring buffer, JPEG ring buffer or RAW data ring buffer
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****Memory Interface Current Offset Counter of Main Picture Y Component JPEG or RAW Register****MI\_MP\_Y\_OFFSET\_CNT\_SHD****Memory Interface Current Offset Counter of Main Picture Y Component JPEG or RAW Register(1580<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

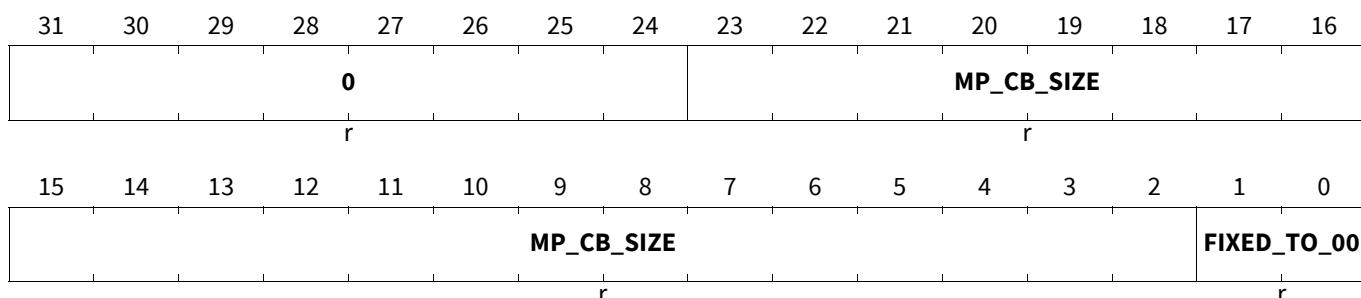
Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_Y_OFFSET_C NT</b>	23:2	r	<b>Main Picture Y Offset Counter</b> Current offset counter of main picture Y component, JPEG or RAW data ringbuffer for address generation  <i>Note:</i> Soft reset will reset the contents to reset value.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

**Memory Interface Shadow Register of Fill Level Interrupt Offset Value For Main Picture Y Register****MI\_MP\_Y\_IRQ\_OFFSET\_SHD****Memory Interface Shadow Register of Fill Level Interrupt Offset Value For Main Picture Y Register(1584<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

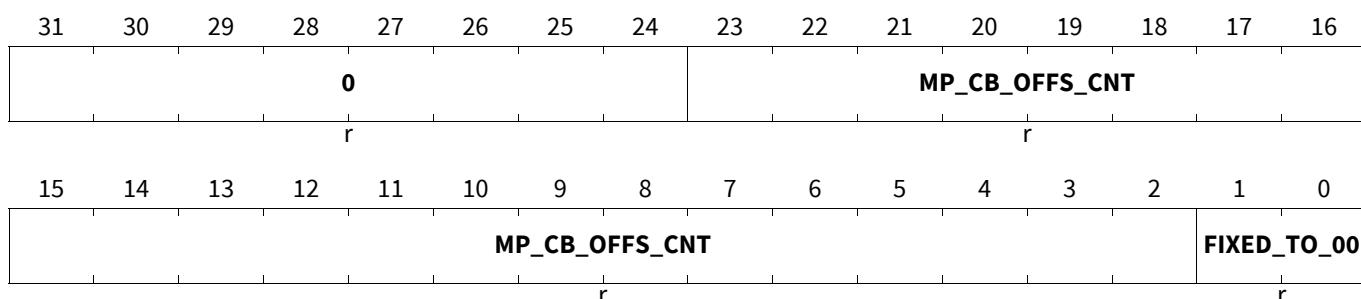
Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_Y_IRQ_OF FS</b>	23:2	r	<b>Main Picture Y IRQ Offset</b> Reaching this offset value by the current offset counter for addressing main picture Y component, JPEG or RAW data leads to generation of fill level interrupt fill_mp_y.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****Memory Interface Base Address Shadow Register For Main Picture Cb Component Ring Register****MI\_MP\_CB\_BASE\_AD\_SHD****Memory Interface Base Address Shadow Register For Main Picture Cb Component Ring Register(1588<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

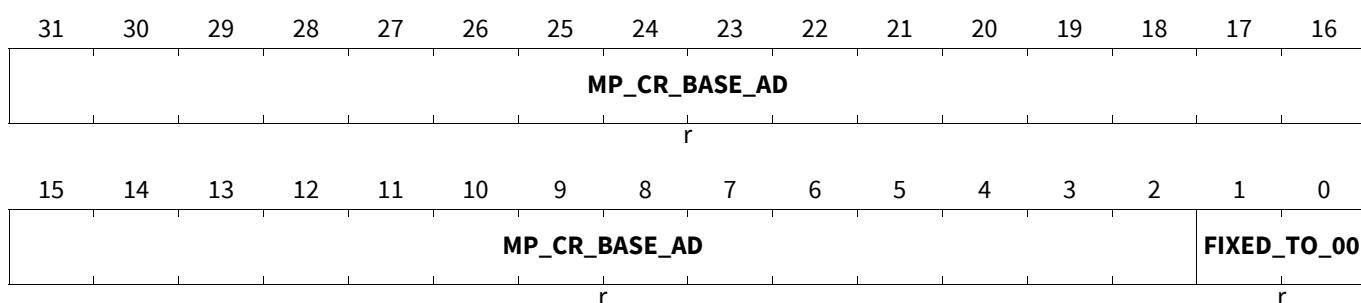
Field	Bits	Type	Description
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value).
MP_CB_BASE _AD	31:2	r	<b>Main Picture Cb Base Address</b> Base address of main picture Cb component ring buffer

**Memory Interface Size Shadow Register Of Main Picture Cb Component Ring Buffer Register****MI\_MP\_CB\_SIZE\_SHD****Memory Interface Size Shadow Register Of Main Picture Cb Component Ring Buffer Register(158C<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

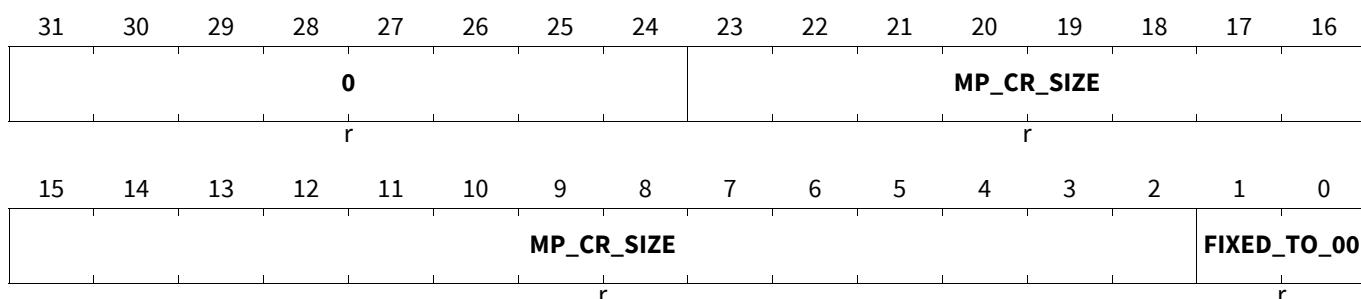
Field	Bits	Type	Description
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value).
MP_CB_SIZE	23:2	r	<b>Main Picture Cb Size</b> Size of main picture Cb component ring buffer
0	31:24	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****Memory Interface Current Offset Counter Of Main Picture Cb Component Ring Buffer Register****MI\_MP\_CB\_OFFSET\_CNT\_SHD****Memory Interface Current Offset Counter Of Main Picture Cb Component Ring Buffer Register( $1590_{H}$ )****Application Reset Value:  $0000\ 0000_{H}$** 

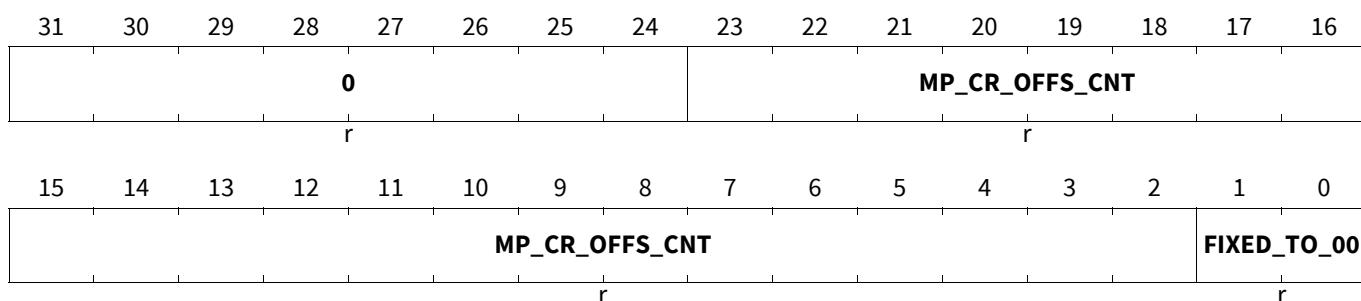
Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_CB_OFFSET_CNT</b>	23:2	r	<b>Main Picture Cb Offset Counter</b> Current offset counter of main picture Cb component ring buffer for address generation  <i>Note:</i> Soft reset will reset the contents to reset value.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

**Memory Interface Base Address Shadow Register For Main Picture Cr Component Ring Register****MI\_MP\_CR\_BASE\_AD\_SHD****Memory Interface Base Address Shadow Register For Main Picture Cr Component Ring Register( $1594_{H}$ )****Application Reset Value:  $0000\ 0000_{H}$** 

Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_CR_BASE_AD</b>	31:2	r	<b>Main Picture Cr Base Address</b> Base address of main picture Cr component ring buffer

**Camera and ADC Interface (CIF)****Memory Interface Size Shadow Register Of Main Picture Cr Component Ring Buffer Register****MI\_MP\_CR\_SIZE\_SHD****Memory Interface Size Shadow Register Of Main Picture Cr Component Ring Buffer Register(1598<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_CR_SIZE</b>	23:2	r	<b>Main Picture Cr Size</b> Size of main picture Cr component ring buffer
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

**Memory Interface Current Offset Counter Of Main Picture Cr Component Ring Buffer Register****MI\_MP\_CR\_OFFSET\_CNT\_SHD****Memory Interface Current Offset Counter Of Main Picture Cr Component Ring Buffer Register(159C<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

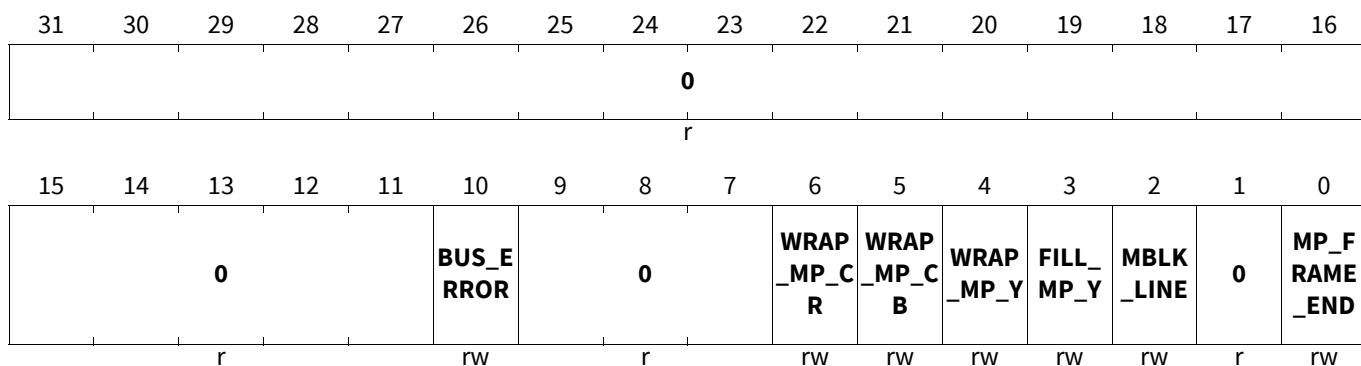
Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>MP_CR_OFFSET_CNT</b>	23:2	r	<b>Main Picture Cr Offset Counter</b> Current offset counter of main picture Cr component ring buffer for address generation  <i>Note:</i> Soft reset will reset the contents to reset value.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

## Camera and ADC Interface (CIF)

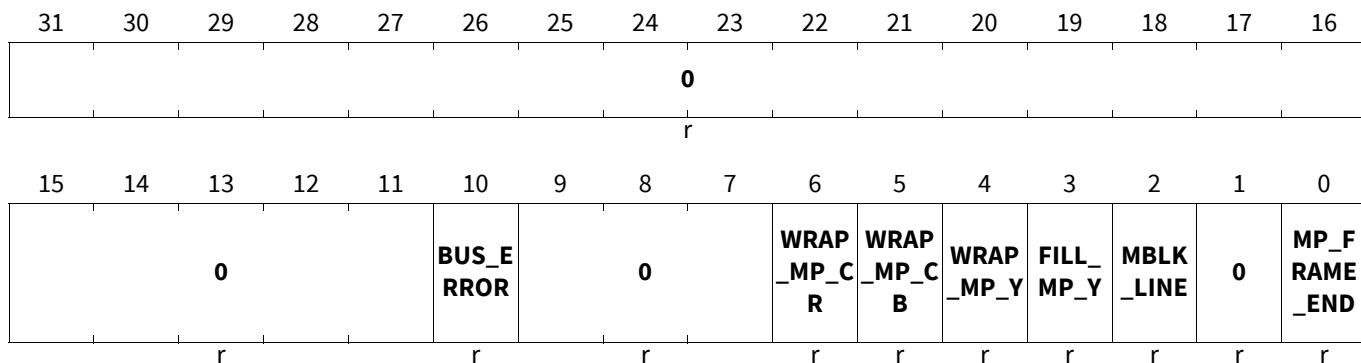
## 26.4.4.3 Memory Interface Interrupt Registers

MI Interrupt Mask ‘1’ interrupt active ‘0’ interrupt masked

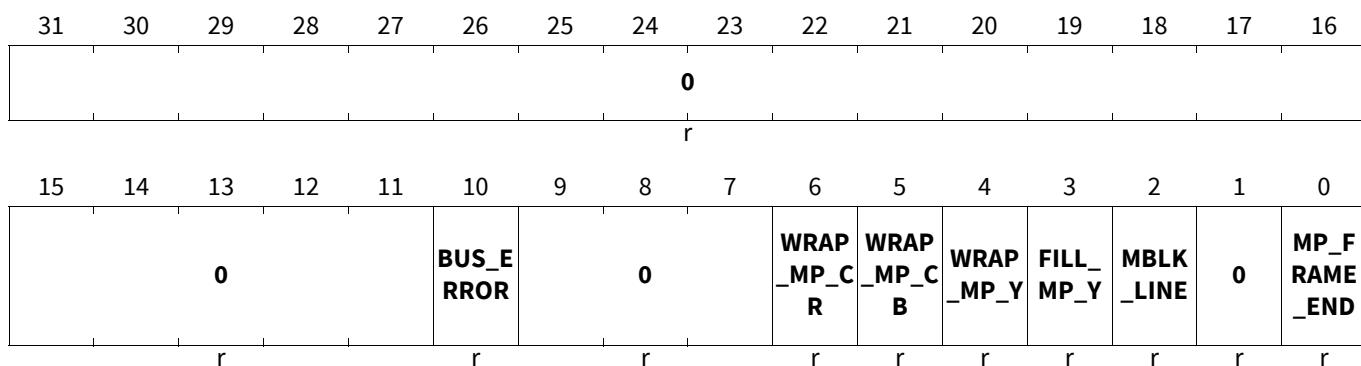
## MI\_IMSC

MI Interrupt Mask ‘1’ interrupt active ‘0’ interrupt masked(15F8<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>

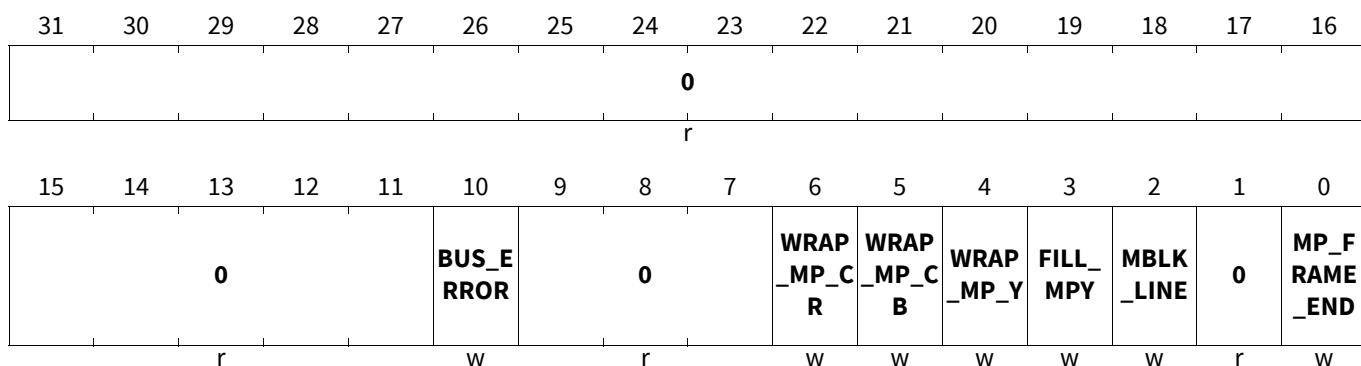
Field	Bits	Type	Description
MP_FRAME_E ND	0	rw	<b>Main Picture Frame End</b> Mask main picture end of frame interrupt0
MBLK_LINE	2	rw	<b>Macro Block Line Interrupt</b> Mask bit for macroblock line interrupt of main picture (16 lines of Y, 8 lines of Cb and 8 lines of Cr are written into RAM)
FILL_MP_Y	3	rw	<b>Fill Main Picture Y</b> Mask bit for fill level interrupt of main picture Y, JPEG or RAW data
WRAP_MP_Y	4	rw	<b>Wrap Main Picture Y</b> Mask bit for main picture Y address wrap interrupt
WRAP_MP_CB	5	rw	<b>Wrap Main Picture Cb</b> Mask bit for main picture Cb address wrap interrupt
WRAP_MP_CR	6	rw	<b>Wrap Main Picture Cr</b> Mask bit for main picture Cr address wrap interrupt
BUS_ERROR	10	rw	<b>Bus Error</b> Mask bit for Bus write or read error interrupt (from handshake target interfaces)
0	1, 9:7, 31:11	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****MI Raw Interrupt Status Register****MI\_RIS****MI Raw Interrupt Status Register****(15FC<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

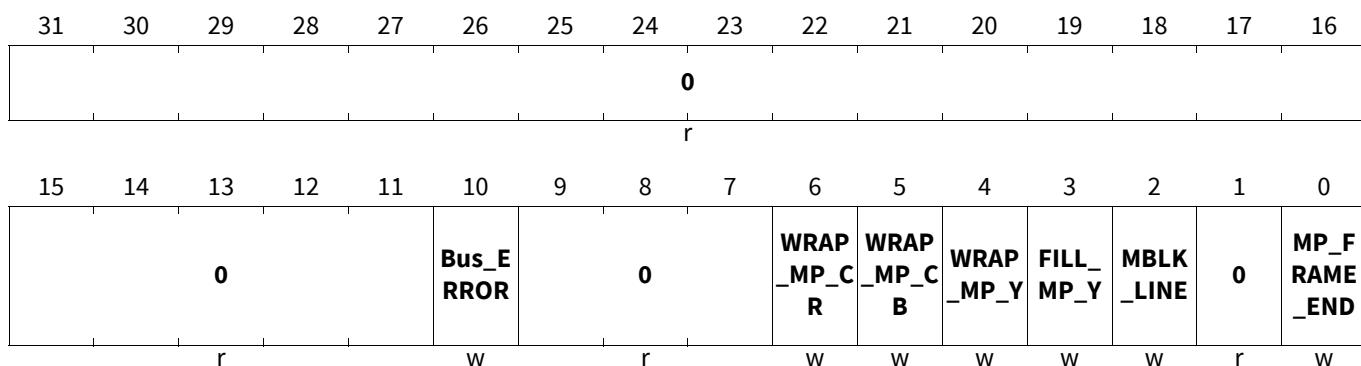
Field	Bits	Type	Description
<b>MP_FRAME_E ND</b>	0	r	<b>Main Picture Frame End</b> Raw status of main picture end of frame interrupt
<b>MBLK_LINE</b>	2	r	<b>Macro Block Line Interrupt</b> Raw status of macroblock line interrupt of main picture (16 lines of Y, 8 lines of Cb and 8 lines of Cr are written into RAM, valid only for planar and semi-planar mode)
<b>FILL_MP_Y</b>	3	r	<b>Fill Main Picture Y</b> Raw status of fill level interrupt of main picture Y, JPEG or RAW data
<b>WRAP_MP_Y</b>	4	r	<b>Wrap Main Picture Y</b> Raw status of main picture Y address wrap interrupt
<b>WRAP_MP_CB</b>	5	r	<b>Wrap Main Picture Cb</b> Raw status of main picture Cb address wrap interrupt
<b>WRAP_MP_CR</b>	6	r	<b>Wrap Main Picture Cr</b> Raw status of main picture Cr address wrap interrupt
<b>BUS_ERROR</b>	10	r	<b>Bus Error</b> Raw status of Bus write or read error interrupt (from handshake target interfaces)
<b>0</b>	1, 9:7, 31:11	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****MI Masked Interrupt Status Register****MI\_MIS**
**MI Masked Interrupt Status Register (1600<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**


Field	Bits	Type	Description
<b>MP_FRAME_E ND</b>	0	r	<b>Main Picture Frame End</b> Masked status of main picture end of frame interrupt
<b>MBLK_LINE</b>	2	r	<b>Macro Block Line Interrupt</b> Masked status of macroblock line interrupt of main picture (16 lines of Y, 8 lines of Cb and 8 lines of Cr are written into RAM, valid only for planar and semi-planar mode)
<b>FILL_MP_Y</b>	3	r	<b>Fill Main Picture Y</b> Masked status of fill level interrupt of main picture Y, JPEG or RAW data
<b>WRAP_MP_Y</b>	4	r	<b>Wrap Main Picture Y</b> Masked status of main picture Y address wrap interrupt
<b>WRAP_MP_CB</b>	5	r	<b>Wrap Main Picture Cb</b> Masked status of main picture Cb address wrap interrupt
<b>WRAP_MP_CR</b>	6	r	<b>Wrap Main Picture Cr</b> Masked status of main picture Cr address wrap interrupt
<b>BUS_ERROR</b>	10	r	<b>Bus Error</b> Masked status of Bus write or read error interrupt (from handshake target interfaces)
<b>0</b>	1, 9:7, 31:11	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****MI Interrupt Clear Register****MI\_ICR****MI Interrupt Clear Register**(1604<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>MP_FRAME_E ND</b>	0	w	<b>Main Picture Frame End</b> Clear main picture end of frame interrupt
<b>MBLK_LINE</b>	2	w	<b>Macro Block Line Interrupt</b> Clear macroblock line interrupt of main picture (16 lines of Y, 8 lines of Cb and 8 lines of Cr are written into RAM, valid only for planar and semi-planar mode)
<b>FILL_MPY</b>	3	w	<b>Fill Main Picture Y</b> Clear fill level interrupt of main picture Y, JPEG or RAW data
<b>WRAP_MP_Y</b>	4	w	<b>Wrap Main Picture Y</b> Clear main picture Y address wrap interrupt
<b>WRAP_MP_CB</b>	5	w	<b>Wrap Main Picture Cb</b> Clear main picture Cb address wrap interrupt
<b>WRAP_MP_CR</b>	6	w	<b>Wrap Main Picture Cr</b> Clear main picture Cr address wrap interrupt
<b>BUS_ERROR</b>	10	w	<b>Bus Error</b> Clear Bus write or read error interrupt (from handshake target interfaces)
<b>0</b>	1, 9:7, 31:11	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****MI Interrupt Set Register****MI\_ISR****MI Interrupt Set Register****(1608<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>MP_FRAME_E_ND</b>	0	w	<b>Main Picture Frame End</b> Set main picture end of frame interrupt
<b>MBLK_LINE</b>	2	w	<b>Macro Block Line Interrupt</b> Set macroblock line interrupt of main picture (16 lines of Y, 8 lines of Cb and 8 lines of Cr are written into RAM, valid only for planar and semi-planar mode)
<b>FILL_MP_Y</b>	3	w	<b>Fill Main Picture Y</b> Set fill level interrupt of main picture Y, JPEG or RAW data
<b>WRAP_MP_Y</b>	4	w	<b>Wrap Main Picture Y</b> Set main picture Y address wrap interrupt
<b>WRAP_MP_CB</b>	5	w	<b>Wrap Main Picture Cb</b> Set main picture Cb address wrap interrupt
<b>WRAP_MP_CR</b>	6	w	<b>Wrap Main Picture Cr</b> Set main picture Cr address wrap interrupt
<b>Bus_ERROR</b>	10	w	<b>Bus Error</b> Set Bus write or read error interrupt (from handshake target interfaces)
<b>0</b>	1, 9:7, 31:11	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****MI Status Register****MI\_STATUS****MI Status Register**(160C<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16				
0																			
r																			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
0								BUS_WRITE_ERRO_R		0				MP_C_R_FIF_O_FUL_L		MP_C_B_FIF_O_FUL_L		MP_Y_FIFO_FULL	
r								r		r				r		r		r	

Field	Bits	Type	Description
MP_Y_FIFO_FULL	0	r	<b>Main Picture Y FIFO Full</b> FIFO full flag of Y FIFO in main path asserted since last clear
MP_CB_FIFO_FULL	1	r	<b>Main Picture Cb FIFO Full</b> FIFO full flag of Cb FIFO in main path asserted since last clear
MP_CR_FIFO_FULL	2	r	<b>Main Picture Cr FIFO Full</b> FIFO full flag of Cr FIFO in main path asserted since last clear
BUS_WRITE_E_RROR	8	r	<b>Bus Write Error</b> An Bus error occurred (Bus_error interrupt raised) while writing to the Bus (main/self path) since last clear.
0	7:3, 31:9	r	<b>Reserved</b> Read as 0.

**MI Status Clear Register****MI\_STATUS\_CLR****MI Status Clear Register**(1610<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16				
0																			
r																			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
0								BUS_WRITE_ERRO_R		0				MP_C_R_FIF_O_FUL_L		MP_C_B_FIF_O_FUL_L		MP_Y_FIFO_FULL	
r								w		r				w		w		w	

## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
<b>MP_Y_FIFO_FULL</b>	0	w	<b>Main Picture Y FIFO Full</b> Clear status of Y FIFO full flag in main path
<b>MP_CB_FIFO_FULL</b>	1	w	<b>Main Picture Cb FIFO Full</b> Clear status of Cb FIFO full flag in main path
<b>MP_CR_FIFO_FULL</b>	2	w	<b>Main Picture Cr FIFO Full</b> Clear status of Cr FIFO full flag in main path
<b>BUS_WRITE_ERROR</b>	8	w	<b>Bus Write Error</b> Clear status of Bus write error flag
<b>EP_1_FIFO_FULL</b>	24	w	<b>Extra Path 1 FIFO Full</b> Clear status of FIFO full flag in extra path
<b>EP_2_FIFO_FULL</b>	25	w	<b>Extra Path 2 FIFO Full</b> Clear status of FIFO full flag in extra path
<b>EP_3_FIFO_FULL</b>	26	w	<b>Extra Path 3 FIFO Full</b> Clear status of FIFO full flag in extra path
<b>EP_4_FIFO_FULL</b>	27	w	<b>Extra Path 4 FIFO Full</b> Clear status of FIFO full flag in extra path
<b>EP_5_FIFO_FULL</b>	28	w	<b>Extra Path 5 FIFO Full</b> Clear status of FIFO full flag in extra path
<b>0</b>	7:3, 23:9, 31:29	r	<b>Reserved</b> Read as 0, should be written with 0.

## Camera and ADC Interface (CIF)

## 26.4.5 JPEG Encoder Programming Registers

The address of each CIF JPEG encoder register is evaluated as CIF\_JPE\_BASE + Offset.

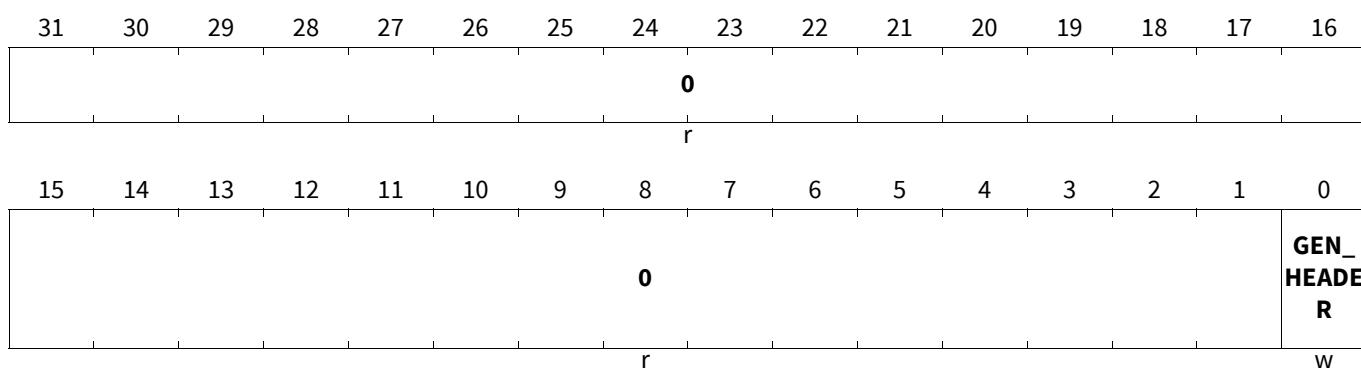
## 26.4.5.1 JPEG Encoder Control Registers

## JPE Command To Start Stream Header Generation Register

Register 305

## JPE\_GEN\_HEADER

JPE Command To Start Stream Header Generation Register( $1900_{H}$ ) Application Reset Value:  $0000\ 0000_{H}$

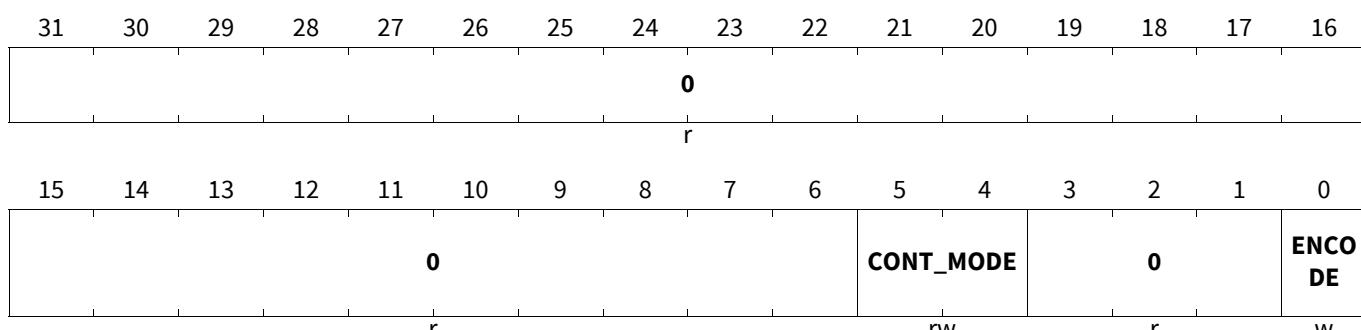


Field	Bits	Type	Description
GEN_HEADER	0	w	<b>Generate Header</b> $1_B$ Start command to generate stream header; auto reset to zero after one clock cycle
0	31:1	r	<b>Reserved</b> Read as 0, should be written with 0.

## JPE Start Command To Start JFIF Stream Encoding Register

## JPE\_ENCODE

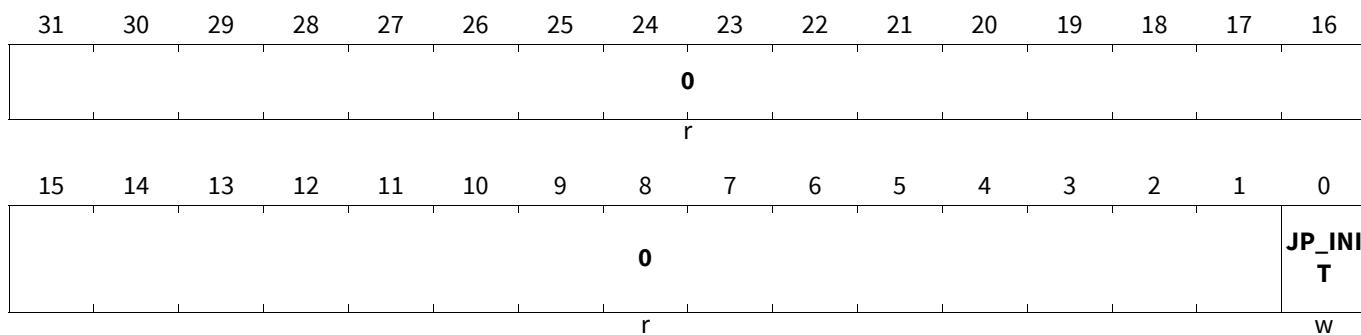
JPE Start Command To Start JFIF Stream Encoding Register( $1904_{H}$ ) Application Reset Value:  $0000\ 0000_{H}$



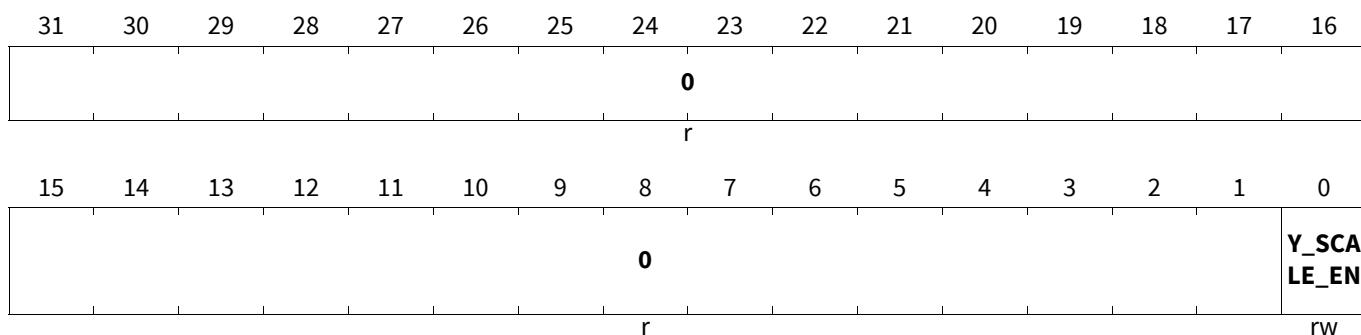
Field	Bits	Type	Description
ENCODE	0	w	<b>Encode</b> $1_B$ Start command to start JFIF stream encoding; auto reset to zero after one clock cycle

**Camera and ADC Interface (CIF)**

Field	Bits	Type	Description
<b>CONT_MODE</b>	5:4	rw	<b>Encoder continuous mode</b> "00": encoder stops at frame end (corresponds to former behavior) "01": encoder starts automatically to encode the next frame "10": unused "11": encoder first generates next header and then encodes automatically the next frame These settings are checked after encoding one frame. They are not auto-reset by hardware.
<b>0</b>	31:1, 31:6	r	<b>Reserved</b> Read as 0, should be written with 0.

**JPE Automatic Configuration Update Register****JPE\_INIT****JPE Automatic Configuration Update Register (1908<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>JP_INIT</b>	0	w	<b>JPEG Init</b> This bit has to be set after "Encode" to start the JPEG encoder. The "Encode" command becomes active either with JP_INIT or with the input signal "CFG_UPD" auto reset to zero after one clock cycle !!! <sub>1_B</sub> Immediate start of JPEG encoder.
<b>0</b>	31:1	r	<b>Reserved</b> Read as 0, should be written with 0.

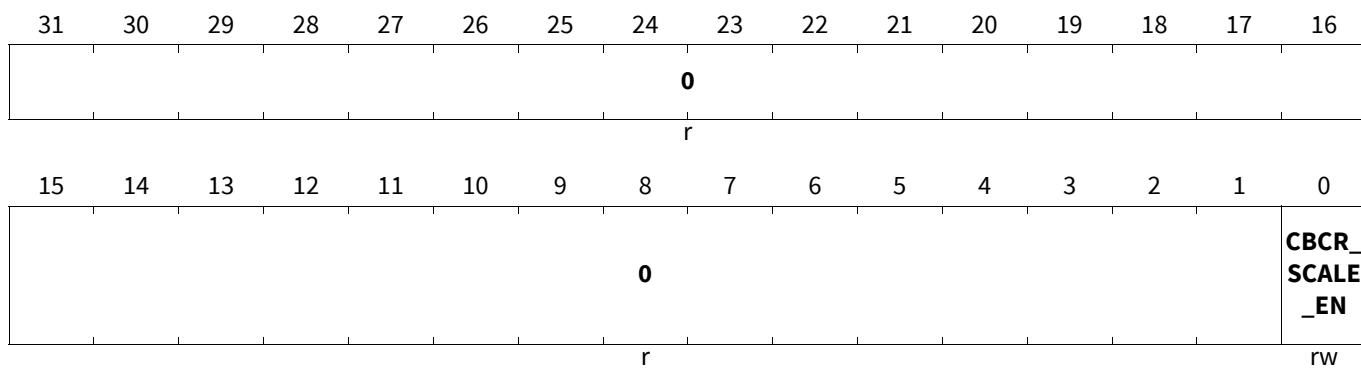
**JPE Y Value Scaling Control Register****JPE\_Y\_SCALE\_EN****JPE Y Value Scaling Control Register****(190C<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
<b>Y_SCALE_EN</b>	0	rw	<b>Y scale flag</b> $0_B$ no Y input scaling $1_B$ scaling Y input from [16..235] to [0..255]
<b>0</b>	31:1	r	<b>Reserved</b> Read as 0, should be written with 0.

## JPE Cb/Cr Value Scaling Control Register

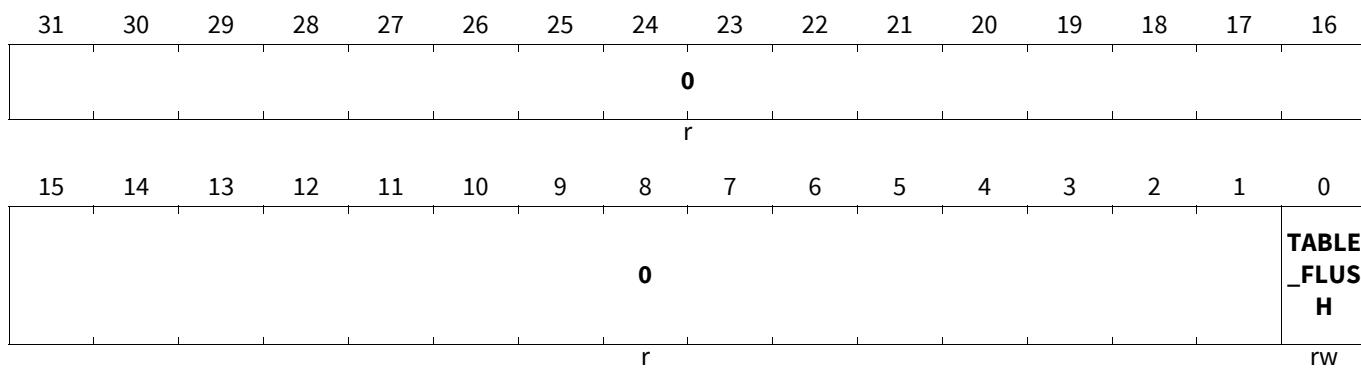
## JPE\_CBCR\_SCALE\_EN

JPE Cb/Cr Value Scaling Control Register (1910<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>CBCR_SCALE_EN</b>	0	rw	<b>Cb/Cr scale flag</b> $0_B$ no Cb/Cr input scaling $1_B$ scaling Cb/Cr input from [16..240] to [0..255]
<b>0</b>	31:1	r	<b>Reserved</b> Read as 0, should be written with 0.

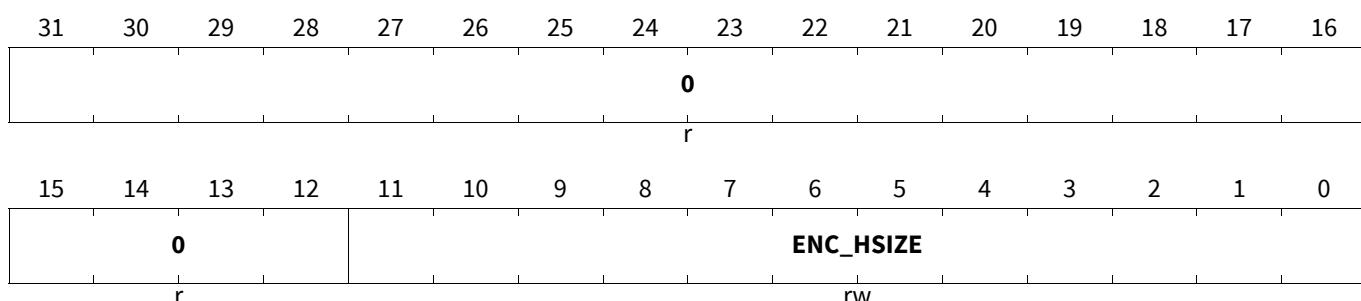
## JPE Header Generation Debug Register

## JPE\_TABLE\_FLUSH

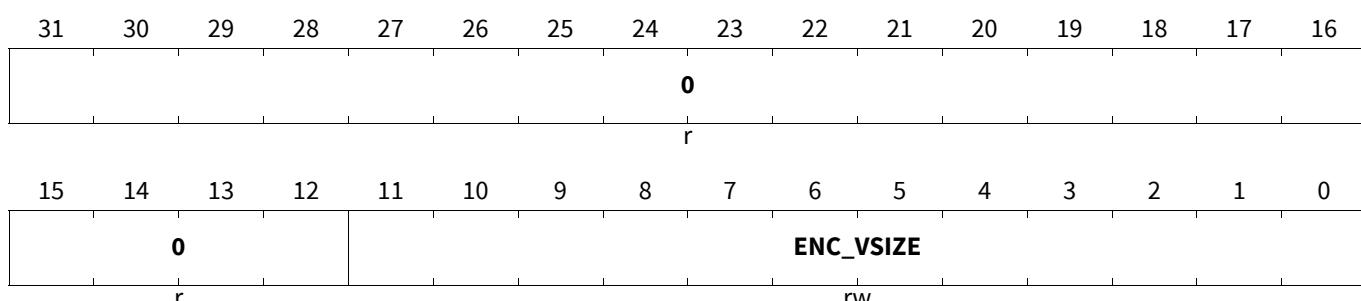
JPE Header Generation Debug Register (1914<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>

**Camera and ADC Interface (CIF)**

Field	Bits	Type	Description
<b>TABLE_FLUSH</b>	0	rw	<b>Header generation debug control flag</b> (controls transmission of last header bytes if the 64 bit output buffer is not completely filled) 0 <sub>B</sub> wait for encoded image data to fill output buffer 1 <sub>B</sub> immediately transmit last header bytes
<b>0</b>	31:1	r	<b>Reserved</b> Read as 0, should be written with 0.

**JPEG Codec Horizontal Image Size For Encoding Register****JPE\_ENC\_HSIZE****JPEG Codec Horizontal Image Size For Encoding Register(1918<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>ENC_HSIZE</b>	11:0	rw	<b>Horizontal Size</b> JPEG codec horizontal image size for R2B and SGEN blocks
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**JPEG Codec Vertical Image Size For Encoding Register****JPE\_ENC\_VSIZE****JPEG Codec Vertical Image Size For Encoding Register(191C<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

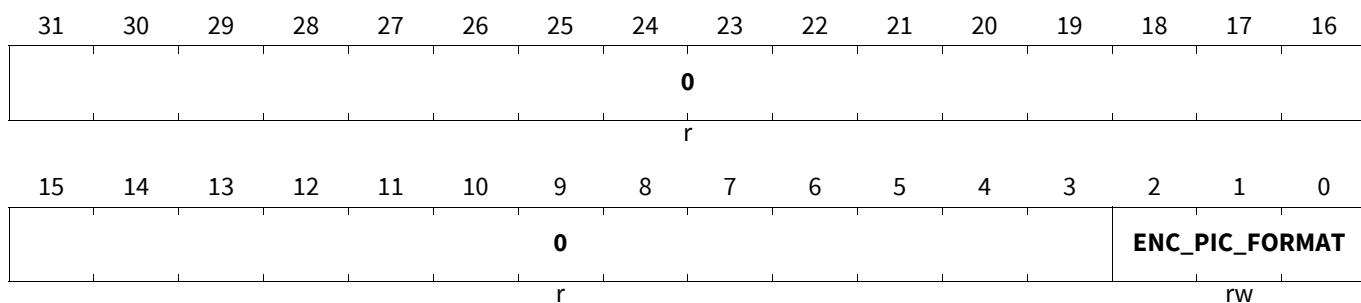
Field	Bits	Type	Description
<b>ENC_VSIZE</b>	11:0	rw	<b>Vertical Size</b> JPEG codec vertical image size for R2B and SGEN blocks

**Camera and ADC Interface (CIF)**

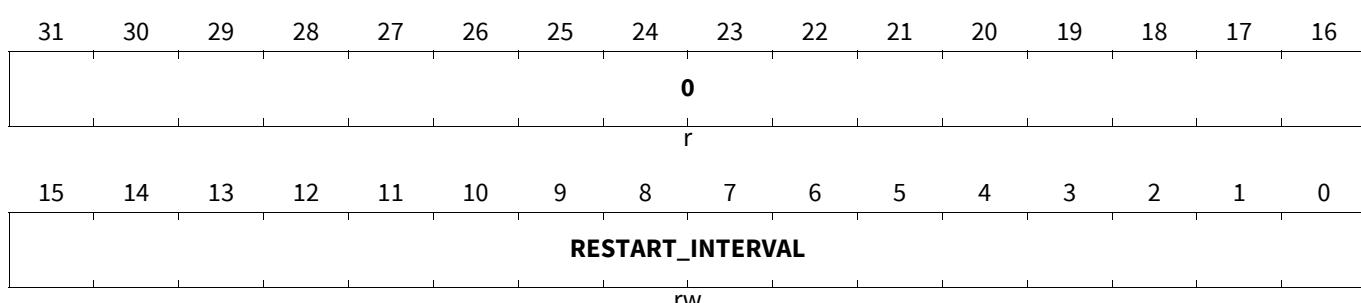
Field	Bits	Type	Description
0	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**JPEG Picture Encoding Format Register**

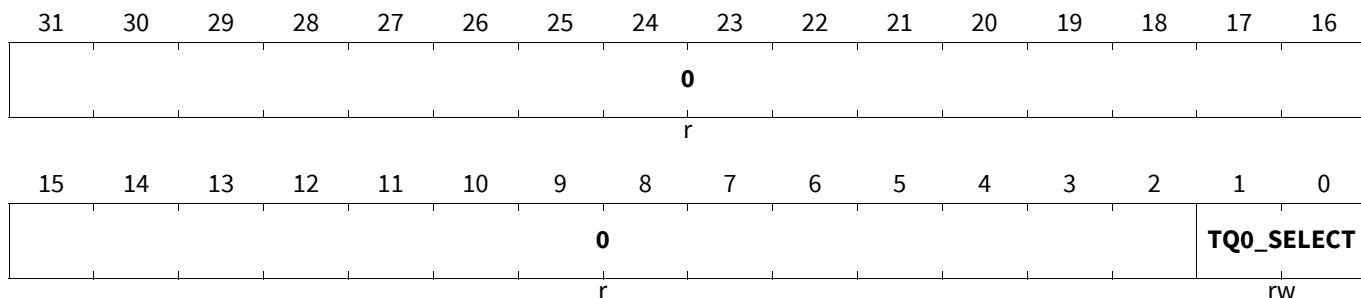
Register 313

**JPE\_PIC\_FORMAT****JPEG Picture Encoding Format Register (1920<sub>H</sub>) Application Reset Value: 0000 0001<sub>H</sub>**

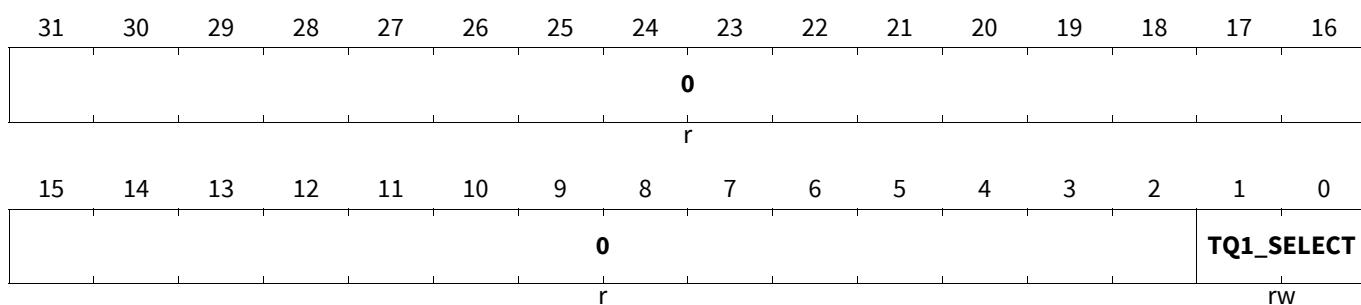
Field	Bits	Type	Description
ENC_PIC_FORMAT	2:0	rw	<b>Picture Encoding Format</b> $001_B$ 4:2:2 format $100_B$ 4:0:0 format ... $111_B$ 4:0:0 format
0	31:3	r	<b>Reserved</b> Read as 0, should be written with 0.

**JPE Restart Marker Insertion Register****JPE\_RESTART\_INTERVAL****JPE Restart Marker Insertion Register (1924<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

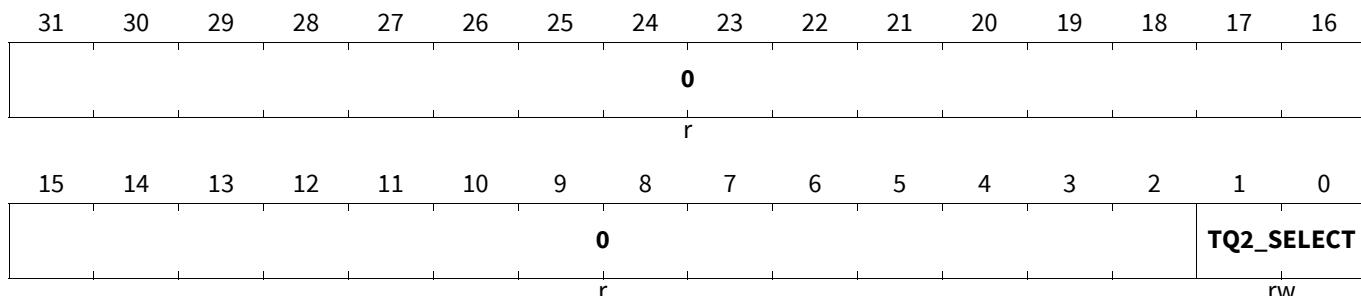
Field	Bits	Type	Description
RESTART_INTERVAL	15:0	rw	<b>Restart Interval</b> No of MCU in reset interval via host
0	31:16	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****Q-table Selector 0, Quant. Table For Y Component****JPE\_TQ\_Y\_SELECT****Q-table Selector 0, Quant. Table For Y Component( $1928_H$ )****Application Reset Value: 0000 0000<sub>H</sub>**

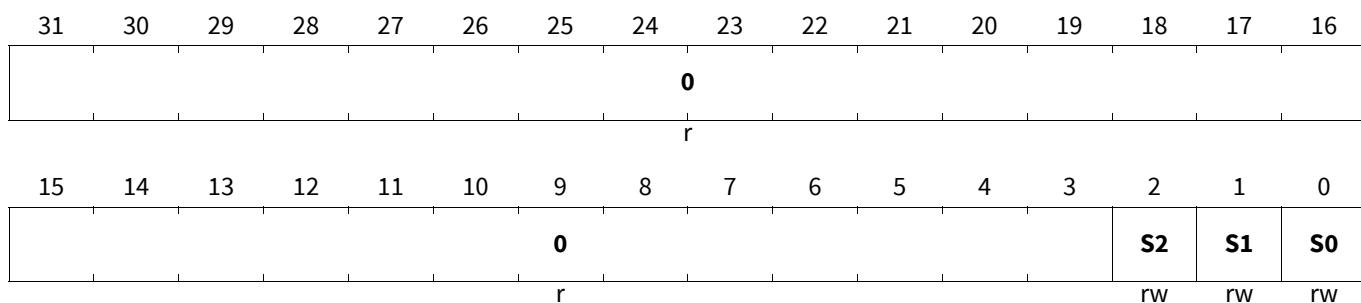
Field	Bits	Type	Description
TQ0_SELECT	1:0	rw	<b>Q-Table Selector Y</b> Q-table selector 0, Quant. For Y component. $00_B$ Q-Table 0 $01_B$ Q-Table 1 $10_B$ Q-Table 2 $11_B$ Q-Table 3
0	31:2	r	<b>Reserved</b> Read as 0, should be written with 0.

**Q-table Selector 1, Quant. Table For U Component****JPE\_TQ\_U\_SELECT****Q-table Selector 1, Quant. Table For U Component( $192C_H$ )****Application Reset Value: 0000 0001<sub>H</sub>**

Field	Bits	Type	Description
TQ1_SELECT	1:0	rw	<b>Q-Table Selector U</b> Q-table selector 1, Quant. For U component. $00_B$ Q-Table 0 $01_B$ Q-Table 1 $10_B$ Q-Table 2 $11_B$ Q-Table 3
0	31:2	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****Q-table Selector 2 Quant Table For V Component****JPE\_TQ\_V\_SELECT****Q-table Selector 2 Quant Table For V Component( $1930_H$ )****Application Reset Value: 0000 0001<sub>H</sub>**

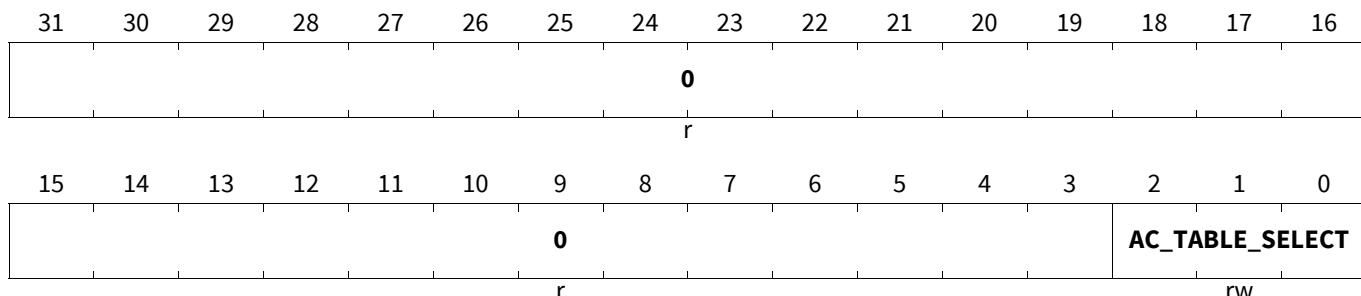
Field	Bits	Type	Description
<b>TQ2_SELECT</b>	1:0	rw	<b>Q-Table Selector V</b> Q-table selector 2, Quant. For V component. 00 <sub>B</sub> Q-Table 0 01 <sub>B</sub> Q-Table 1 10 <sub>B</sub> Q-Table 2 11 <sub>B</sub> Q-Table 3
<b>0</b>	31:2	r	<b>Reserved</b> Read as 0, should be written with 0.

**JPE Huffman Table Selector For DC Values Register****JPE\_DC\_TABLE\_SELECT****JPE Huffman Table Selector For DC Values Register( $1934_H$ )****Application Reset Value: 0000 0006<sub>H</sub>**

Field	Bits	Type	Description
<b>S0</b>	0	rw	<b>DC Table Selector</b> 0 <sub>B</sub> dc table 0; component 0 1 <sub>B</sub> dc table 1; component 0
<b>S1</b>	1	rw	<b>DC Table Selector</b> 0 <sub>B</sub> dc table 0; component 1 1 <sub>B</sub> dc table 1; component 1
<b>S2</b>	2	rw	<b>DC Table Selector</b> 0 <sub>B</sub> dc table 0; component 2 1 <sub>B</sub> dc table 1; component 2

**Camera and ADC Interface (CIF)**

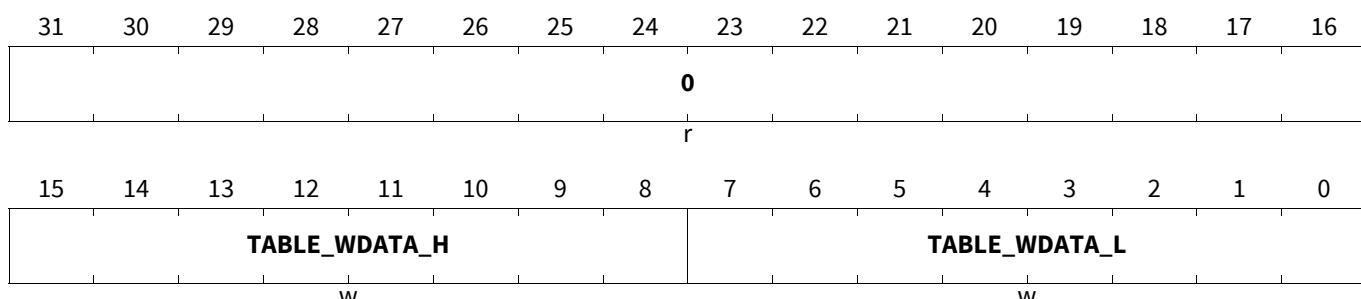
Field	Bits	Type	Description
<b>0</b>	31:3	r	<b>Reserved</b> Read as 0, should be written with 0.

**JPE Huffman Table Selector For AC Values Register****JPE\_AC\_TABLE\_SELECT**JPE Huffman Table Selector For AC Values Register( $1938_{\text{H}}$ )Application Reset Value:  $0000\ 0006_{\text{H}}$ 

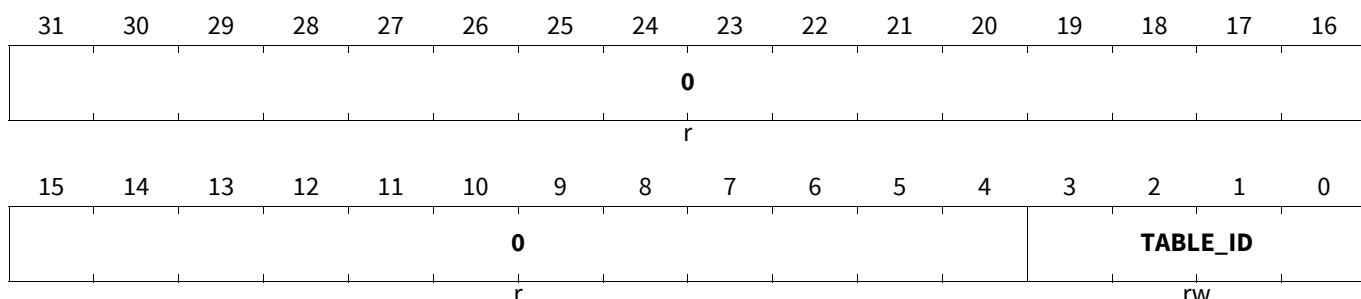
Field	Bits	Type	Description
<b>AC_TABLE_SELECT</b>	2:0	rw	<b>AC Table Selector</b> “xx0” = dc table 0; component 0 “x1x” = dc table 1; component 0 “x0x” = dc table 0; component 1 “x1x” = dc table 1; component 1 “0xx” = dc table 0; component 2 “1xx” = dc table 1; component 2
<b>0</b>	31:3	r	<b>Reserved</b> Read as 0, should be written with 0.

**JPE Table Programming Register****JPE\_TABLE\_DATA**

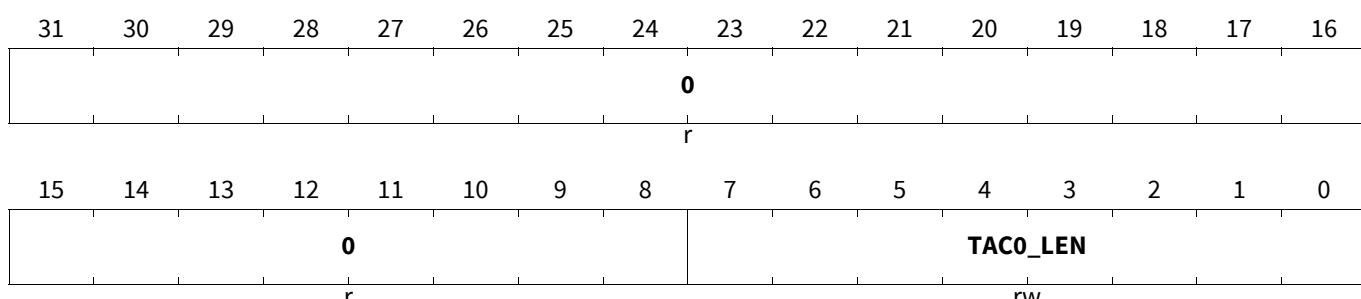
JPE Table Programming Register

 $(193C_{\text{H}})$ Application Reset Value:  $0000\ 0000_{\text{H}}$ 

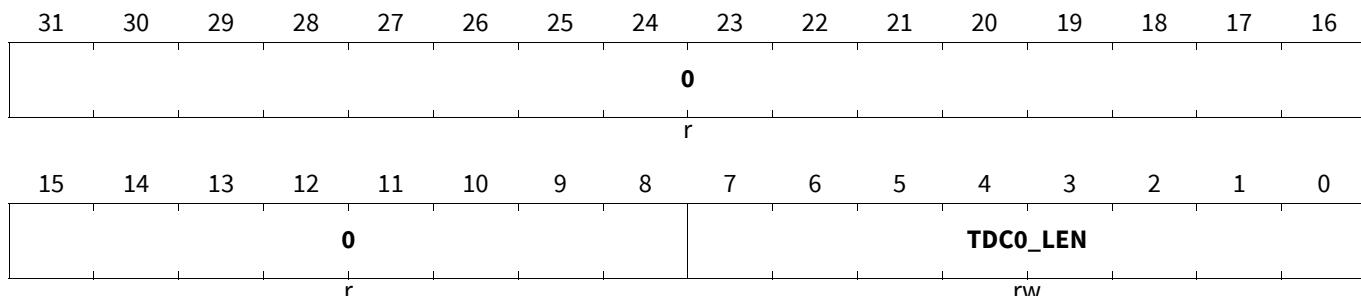
Field	Bits	Type	Description
<b>TABLE_WDAT_A_L</b>	7:0	w	<b>Table data LSB</b>
<b>TABLE_WDAT_A_H</b>	15:8	w	<b>Table data MSB</b>
<b>0</b>	31:16	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****JPE Table Programming Select Register****JPE\_TABLE\_ID**
**JPE Table Programming Select Register (1940<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**


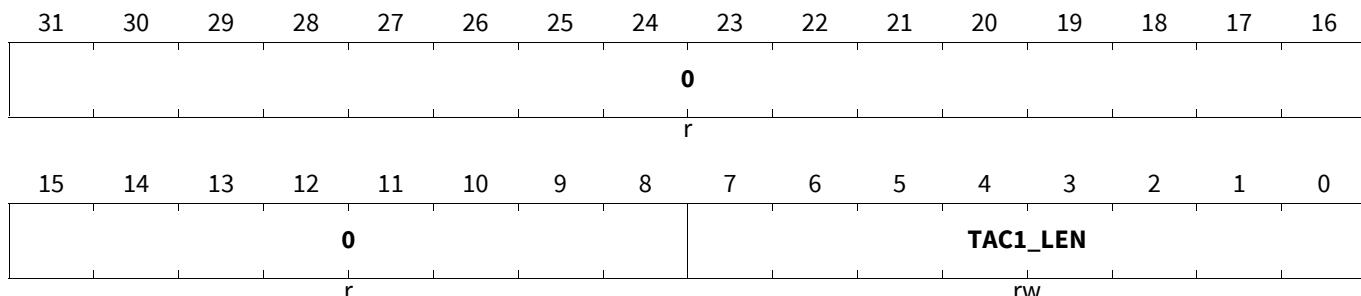
Field	Bits	Type	Description																
<b>TABLE_ID</b>	3:0	rw	<p><b>JPE Table ID</b>  Select table according to the following list. Values 8<sub>H</sub> through F<sub>H</sub> are reserved for debug purposes.</p> <table> <tr><td>0<sub>H</sub></td><td>Q-Table 0</td></tr> <tr><td>1<sub>H</sub></td><td>Q-Table 1</td></tr> <tr><td>2<sub>H</sub></td><td>Q-Table 2</td></tr> <tr><td>3<sub>H</sub></td><td>Q-Table 3</td></tr> <tr><td>4<sub>H</sub></td><td>VLC DC Table 0</td></tr> <tr><td>5<sub>H</sub></td><td>VLC AC Table 0</td></tr> <tr><td>6<sub>H</sub></td><td>VLC DC Table 1</td></tr> <tr><td>7<sub>H</sub></td><td>VLC AC Table 1</td></tr> </table>	0 <sub>H</sub>	Q-Table 0	1 <sub>H</sub>	Q-Table 1	2 <sub>H</sub>	Q-Table 2	3 <sub>H</sub>	Q-Table 3	4 <sub>H</sub>	VLC DC Table 0	5 <sub>H</sub>	VLC AC Table 0	6 <sub>H</sub>	VLC DC Table 1	7 <sub>H</sub>	VLC AC Table 1
0 <sub>H</sub>	Q-Table 0																		
1 <sub>H</sub>	Q-Table 1																		
2 <sub>H</sub>	Q-Table 2																		
3 <sub>H</sub>	Q-Table 3																		
4 <sub>H</sub>	VLC DC Table 0																		
5 <sub>H</sub>	VLC AC Table 0																		
6 <sub>H</sub>	VLC DC Table 1																		
7 <sub>H</sub>	VLC AC Table 1																		
<b>0</b>	31:4	r	<p><b>Reserved</b>  Read as 0, should be written with 0.</p>																

**JPE Huffman AC Table 0 Length Register****JPE\_TAC0\_LEN**
**JPE Huffman AC Table 0 Length Register (1944<sub>H</sub>) Application Reset Value: 0000 00B2<sub>H</sub>**


Field	Bits	Type	Description
<b>TAC0_LEN</b>	7:0	rw	<p><b>AC Table 0 Length</b>  Huffman table length for ac0 table</p>
<b>0</b>	31:8	r	<p><b>Reserved</b>  Read as 0, should be written with 0.</p>

**Camera and ADC Interface (CIF)****JPE Huffman DC Table 0 Length Register****JPE\_TDC0\_LEN****JPE Huffman DC Table 0 Length Register (1948<sub>H</sub>) Application Reset Value: 0000 001C<sub>H</sub>**

Field	Bits	Type	Description
<b>TDC0_LEN</b>	7:0	rw	<b>DC Table 0 Length</b> Huffman table length for dc0 table
<b>0</b>	31:8	r	<b>Reserved</b> Read as 0, should be written with 0.

**JPE Huffman AC Table 1 Length Register****JPE\_TAC1\_LEN****JPE Huffman AC Table 1 Length Register (194C<sub>H</sub>) Application Reset Value: 0000 00B2<sub>H</sub>**

Field	Bits	Type	Description
<b>TAC1_LEN</b>	7:0	rw	<b>AC Table 1 Length</b> Huffman table length for ac1 table
<b>0</b>	31:8	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****JPE Huffman DC Table 1 Length Register****JPE\_TDC1\_LEN****JPE Huffman DC Table 1 Length Register (1950<sub>H</sub>) Application Reset Value: 0000 001C<sub>H</sub>**

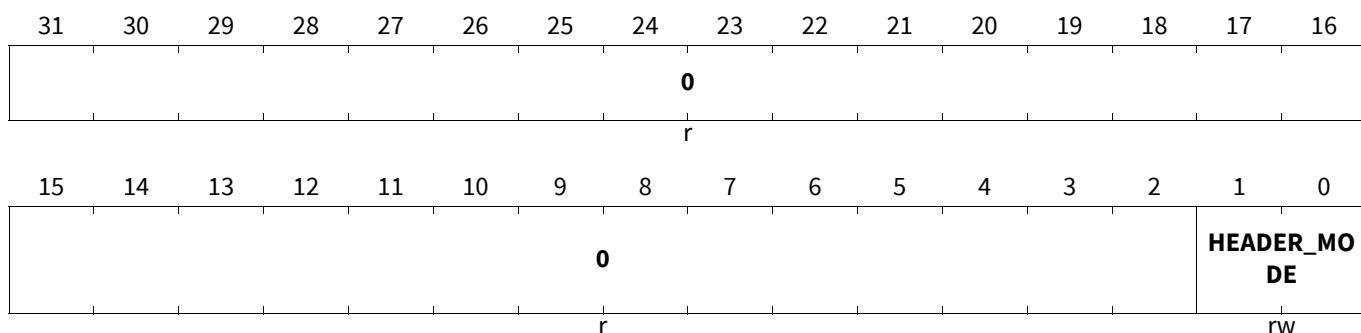
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								TDC1_LEN							
r								rw							

Field	Bits	Type	Description
<b>TDC1_LEN</b>	7:0	rw	<b>DC Table 1 Length</b> Huffman table length for dc1 table
<b>0</b>	31:8	r	<b>Reserved</b> Read as 0, should be written with 0.

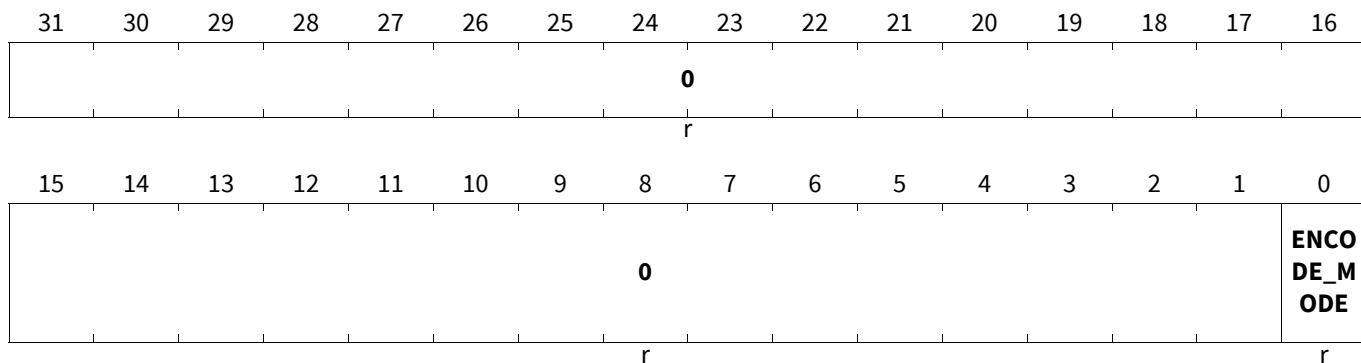
**JPE Encoder Status Flag Register****JPE\_ENCODER\_BUSY****JPE Encoder Status Flag Register (1958<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															r
CODEC_C_BUS_Y															r
r															

Field	Bits	Type	Description
<b>CODEC_BUSY</b>	0	r	<b>Codec Busy</b> 0 <sub>B</sub> Codec is free (not busy) 1 <sub>B</sub> JPEG codec in process
<b>0</b>	31:1	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****JPE Header Mode Definition Register****JPE\_HEADER\_MODE****JPE Header Mode Definition Register**(195C<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
HEADER_MODE	1:0	rw	<b>Header Mode</b> 00 <sub>B</sub> no APPn header 01 <sub>B</sub> don't use this setting 10 <sub>B</sub> JFIF header 11 <sub>B</sub> don't use this setting
0	31:2	r	<b>Reserved</b> Read as 0, should be written with 0.

**JPE Encode Mode Register****JPE\_ENCODE\_MODE****JPE Encode Mode Register**(1960<sub>H</sub>)Application Reset Value: 0000 0001<sub>H</sub>

Field	Bits	Type	Description
ENCODE_MODE	0	r	<b>Encode Mode</b> Always 1, because this is the encoder only edition
0	31:1	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****JPE Debug Information Register****JPE\_DEBUG****JPE Debug Information Register**(1964<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								DEB_B AD_TA BLE_A CCESS	0	DEB_V LC_TA BLE_B USY	DEB_R 2B_ME MORY _FULL	DEB_V LC_EN CODE_ BUSY	DEB_Q IQ_TA BLE_A CC	0	
r								r	r	r	r	r	r	r	r

Field	Bits	Type	Description
<b>DEB_QIQ_TAB</b> <b>LE_ACC</b>	2	r	<b>QIQ Table Access</b> Debug signal only (QIQ table access)
<b>DEB_VLC_ENC</b> <b>ODE_BUSY</b>	3	r	<b>VLC Encode Busy</b> Debug signal only (vlc encode processing active)
<b>DEB_R2B_ME</b> <b>MORY_FULL</b>	4	r	<b>R2B Memory Full</b> Debug signal only (line memory status of r2b)
<b>DEB_VLC_TAB</b> <b>LE_BUSY</b>	5	r	<b>Debug VLC Table Busy</b> Debug signal only (vlc access to huff-tables) Unit will initialize huff tables internally, busy for ~400 cycles after reset.
<b>DEB_BAD_TA</b> <b>BLE_ACCESS</b>	8	r	<b>Debug Bad Table Access</b> Debug signal only (set if an access to the TABLE_DATA or to the TABLE_ID register is performed, when the JPEG_ENCODER is busy. In this case a default handshake acknowledge is generated. Thus the configuration bus is not blocked)
<b>0</b>	1:0, 7:6, 31:9	r	<b>Reserved</b> Read as 0.

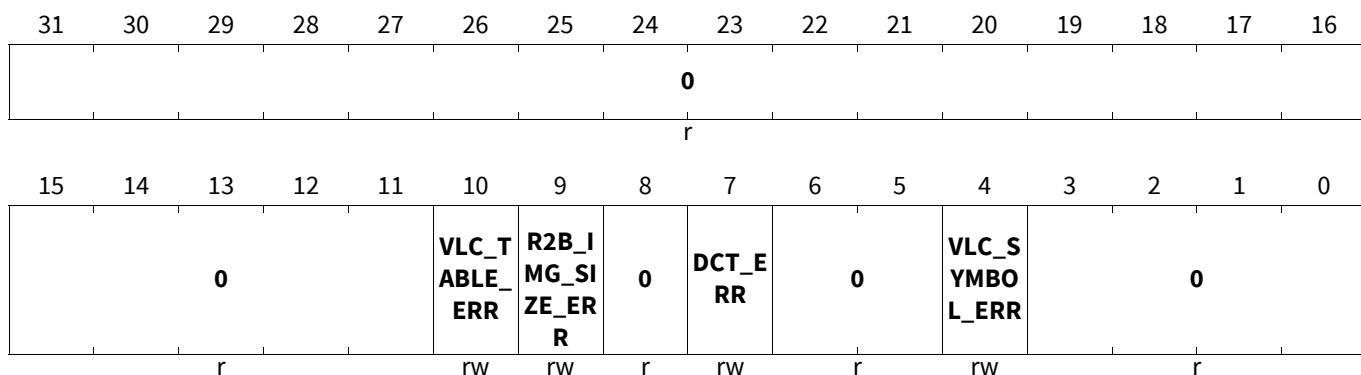
## Camera and ADC Interface (CIF)

## 26.4.5.2 JPEG Encoder Interrupt Registers

## JPE Error Interrupt Mask Register

JPE\_ERROR\_IMR

JPE Error Interrupt Mask Register

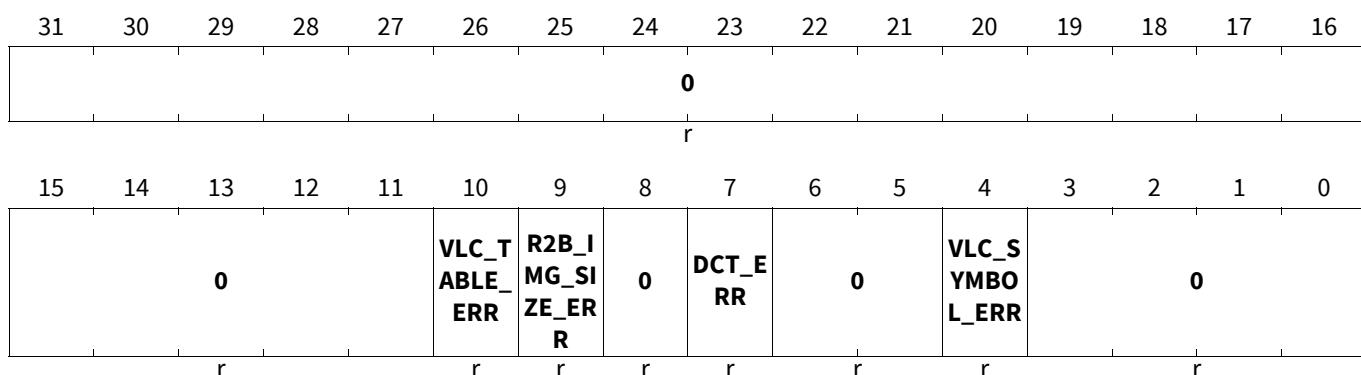
(1968<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
VLC_SYMBOL_ERR	4	rw	<b>VLC Symbol Error</b> 1 <sub>B</sub> interrupt is activated (masked in)
DCT_ERR	7	rw	<b>DC Table Error</b> 1 <sub>B</sub> interrupt is activated (masked in)
R2B_IMG_SIZE_ERR	9	rw	<b>R2B Image Size Error</b> 1 <sub>B</sub> interrupt is activated (masked in)
VLC_TABLE_ERR	10	rw	<b>VLC Table Error</b> 1 <sub>B</sub> interrupt is activated (masked in)
0	3:0, 6:5, 8, 31:11	r	<b>Reserved</b> Read as 0, should be written with 0.

## JPE Error Raw Interrupt Status Register

JPE\_ERROR\_RIS

JPE Error Raw Interrupt Status Register

(196C<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

## Camera and ADC Interface (CIF)

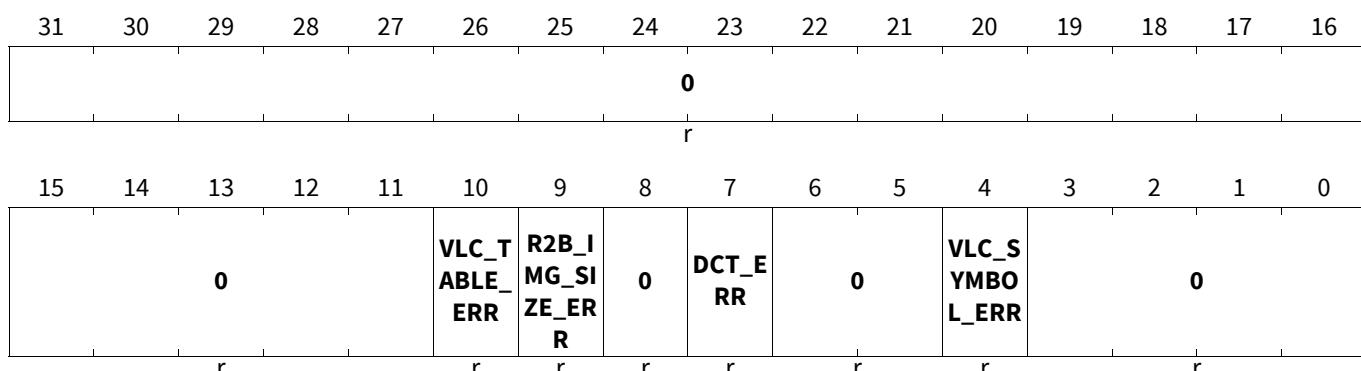
Field	Bits	Type	Description
<b>VLC_SYMBOL_ERR</b>	4	r	<b>VLC Symbol Error</b> 1 <sub>B</sub> illegal symbol detected (encoding)
<b>DCT_ERR</b>	7	r	<b>DC Table Error</b> 1 <sub>B</sub> block start mismatch
<b>R2B_IMG_SIZE_ERR</b>	9	r	<b>R2B Image Size Error</b> 1 <sub>B</sub> mismatch of predefined h_size and v_size values with calculated values (encode mode)
<b>VLC_TABLE_E_RR</b>	10	r	<b>VLC Table Error</b> 1 <sub>B</sub> illegal table detected
0	3:0, 6:5, 8, 31:11	r	<b>Reserved</b> Read as 0

### JPE Error Masked Interrupt Status Register

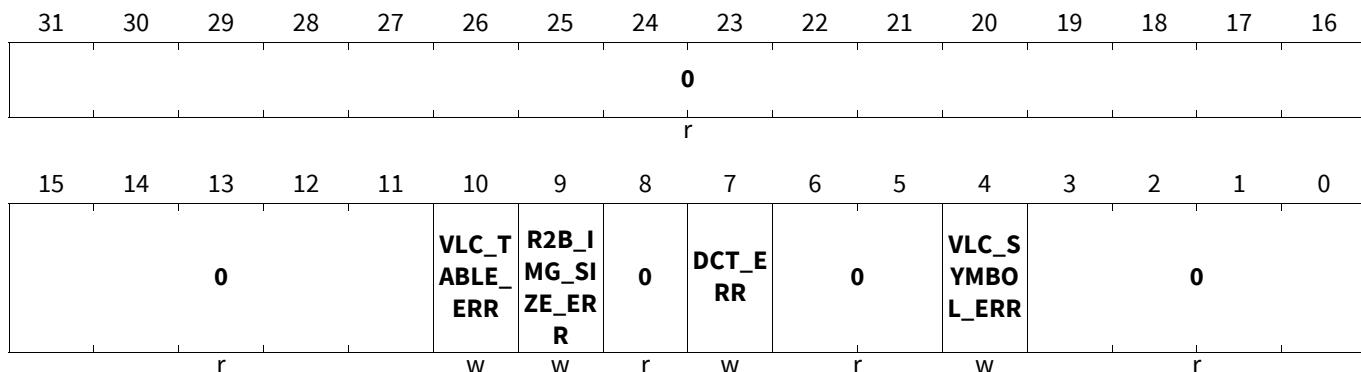
#### JPE\_ERROR\_MIS

JPE Error Masked Interrupt Status Register (1970<sub>H</sub>)

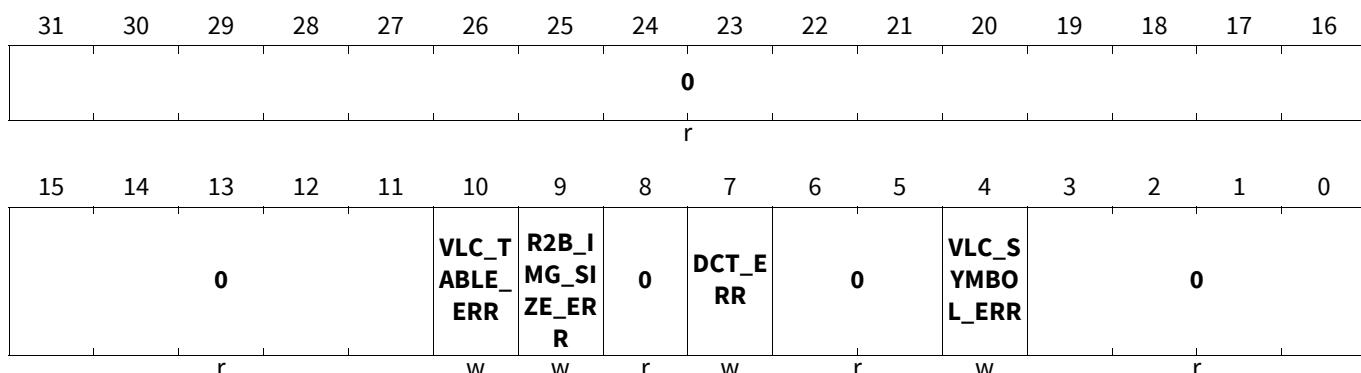
Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>VLC_SYMBOL_ERR</b>	4	r	<b>VLC Symbol Error</b> 1 <sub>B</sub> illegal symbol detected (encoding)
<b>DCT_ERR</b>	7	r	<b>DC Table Error</b> 1 <sub>B</sub> block start mismatch
<b>R2B_IMG_SIZE_ERR</b>	9	r	<b>R2B Image Size Error</b> 1 <sub>B</sub> mismatch of predefined h_size and v_size values with calculated values (encode mode)
<b>VLC_TABLE_E_RR</b>	10	r	<b>VLC Table Error</b> 1 <sub>B</sub> illegal table detected
0	3:0, 6:5, 8, 31:11	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****JPE Error Interrupt Set Register****JPE\_ERROR\_ISR****JPE Error Interrupt Set Register**(1978<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
VLC_SYMBOL_ERR	4	w	<b>VLC Symbol Error</b> 1 <sub>B</sub> set error bit, bit is reset to zero after 1 clk
DCT_ERR	7	w	<b>DC Table Error</b> 1 <sub>B</sub> set error bit, bit is reset to zero after 1 clk
R2B_IMG_SIZE_ERR	9	w	<b>R2B Image Size Error</b> 1 <sub>B</sub> set error bit, bit is reset to zero after 1 clk
VLC_TABLE_ERR	10	w	<b>VLC Table Error</b> 1 <sub>B</sub> set error bit, bit is reset to zero after 1 clk
0	3:0, 6:5, 8, 31:11	r	<b>Reserved</b> Read as 0, should be written with 0.

**JPE Error Interrupt Clear Register****JPE\_ERROR\_ICR****JPE Error Interrupt Clear Register**(1974<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

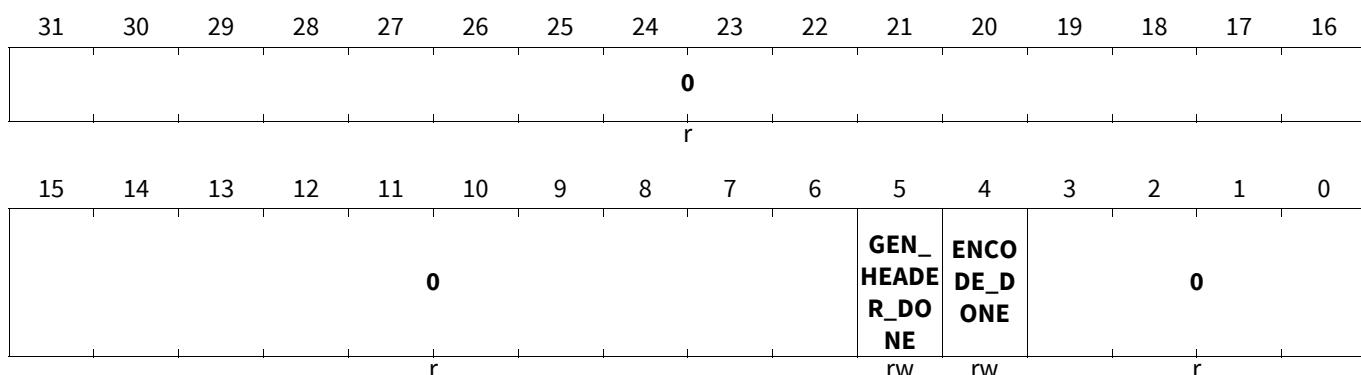
## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
<b>VLC_SYMBOL_ERR</b>	4	w	<b>VLC Symbol Error</b> 1 <sub>B</sub> set error bit, bit is reset to zero after 1 clk
<b>DCT_ERR</b>	7	w	<b>DC Table Error</b> 1 <sub>B</sub> set error bit, bit is reset to zero after 1 clk
<b>R2B_IMG_SIZE_ERR</b>	9	w	<b>R2B Image Size Error</b> 1 <sub>B</sub> set error bit, bit is reset to zero after 1 clk
<b>VLC_TABLE_ERROR</b>	10	w	<b>VLC Table Error</b> 1 <sub>B</sub> set error bit, bit is reset to zero after 1 clk
<b>0</b>	3:0, 6:5, 8, 31:11	r	<b>Reserved</b> Read as 0, should be written with 0.

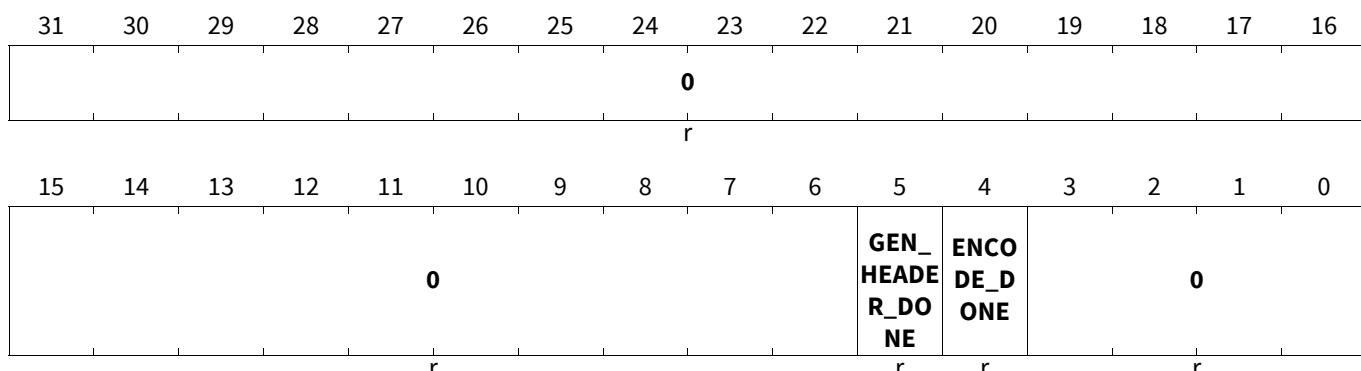
## JPEG Status Interrupt Mask Register

**JPE\_STATUS\_IMR**

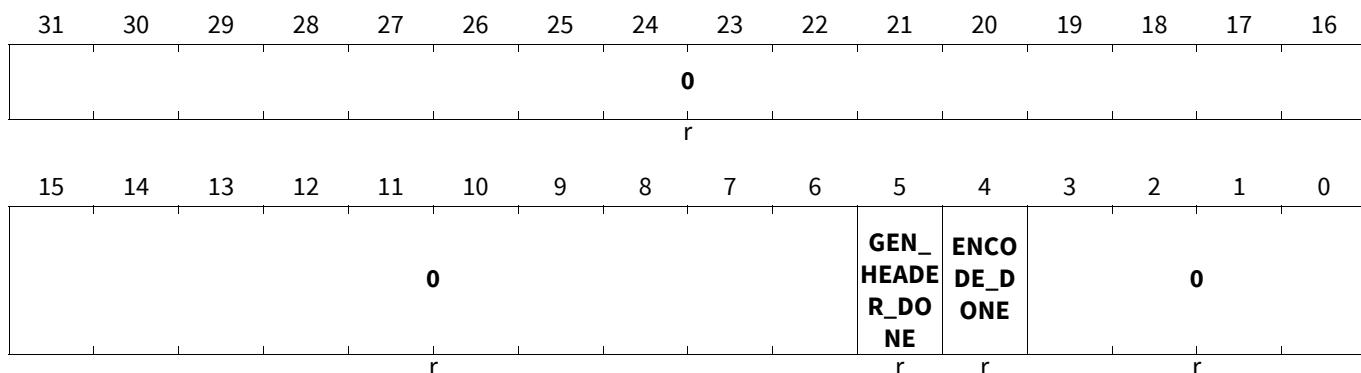
## JPEG Status Interrupt Mask Register

(197C<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>ENCODE_DONE</b>	4	rw	<b>Encoding Complete</b> 1 <sub>B</sub> interrupt is activated (masked in)
<b>GEN_HEADER_DONE</b>	5	rw	<b>Header Generation Complete</b> 1 <sub>B</sub> interrupt is activated (masked in)
<b>0</b>	3:0, 31:6	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****JPEG Status Raw Interrupt Status Register****JPE\_STATUS\_RIS****JPEG Status Raw Interrupt Status Register**(1980<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
ENCODE_DONE	4	r	<b>Encoding Complete</b> 1 <sub>B</sub> Encode processing finished
GEN_HEADER_DONE	5	r	<b>Header Generation Complete</b> 1 <sub>B</sub> Stream header generation finished
0	3:0, 31:6	r	<b>Reserved</b> Read as 0.

**JPEG Status Masked Interrupt Status Register****JPE\_STATUS\_MIS****JPEG Status Masked Interrupt Status Register**Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
ENCODE_DONE	4	r	<b>Encoding Complete</b> 1 <sub>B</sub> Encode processing finished
GEN_HEADER_DONE	5	r	<b>Header Generation Complete</b> 1 <sub>B</sub> Stream header generation finished
0	3:0, 31:6	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****JPEG Status Interrupt Clear Register****JPE\_STATUS\_ICR****JPEG Status Interrupt Clear Register**(1988<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
															0
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															0
r															
															0

Field	Bits	Type	Description
<b>ENCODE_DONE_E</b>	4	w	<b>Encoding Complete</b> 1 <sub>B</sub> clear status bit, bit is reset to zero after 1 clk
<b>GEN_HEADER_DONE</b>	5	w	<b>Header Generation Complete</b> 1 <sub>B</sub> clear status bit, bit is reset to zero after 1 clk
<b>0</b>	3:0, 31:6	r	<b>Reserved</b> Read as 0, should be written with 0.

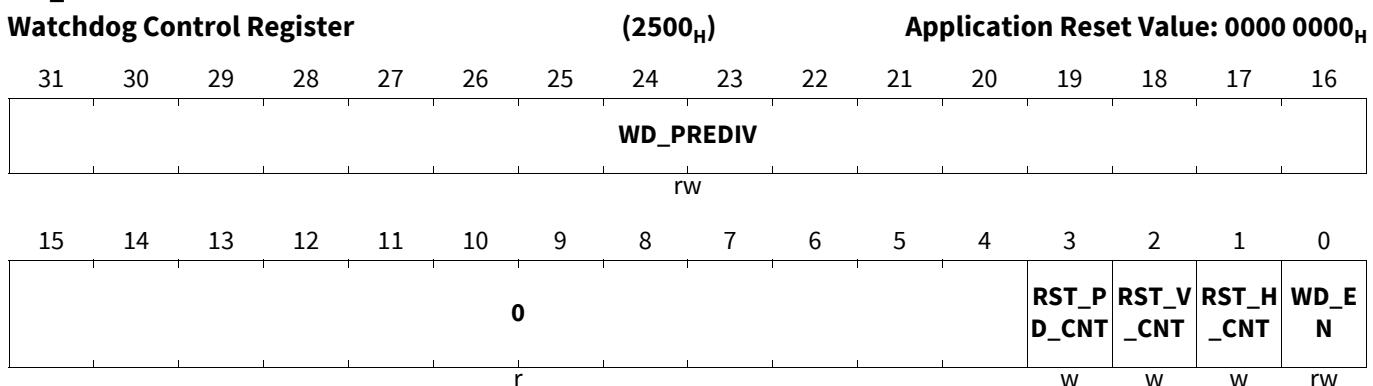
**JPEG Status Interrupt Set Register****JPE\_STATUS\_ISR****JPEG Status Interrupt Set Register**(198C<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
															0
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															0
r															
															0

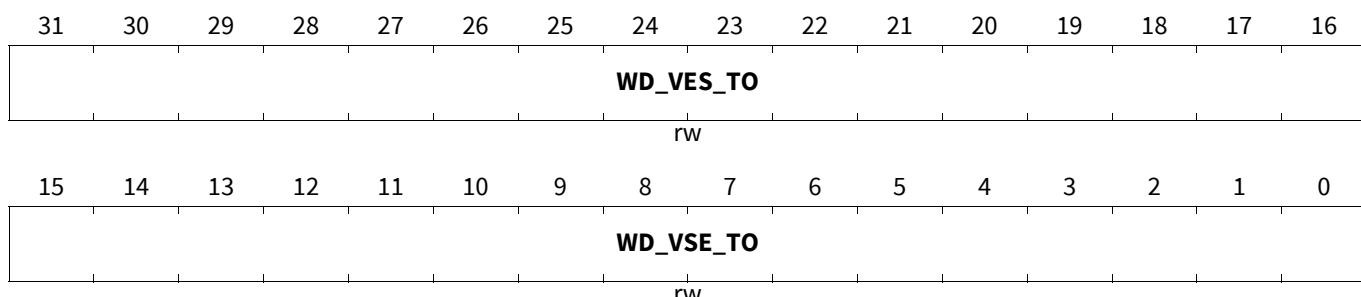
Field	Bits	Type	Description
<b>ENCODE_DONE_E</b>	4	w	<b>Encoding Complete</b> 1 <sub>B</sub> set error bit, bit is reset to zero after 1 clk
<b>GEN_HEADER_DONE</b>	5	w	<b>Header Generation Complete</b> 1 <sub>B</sub> set error bit, bit is reset to zero after 1 clk
<b>0</b>	3:0, 31:6	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****26.4.6 Security Watchdog Programming Registers**

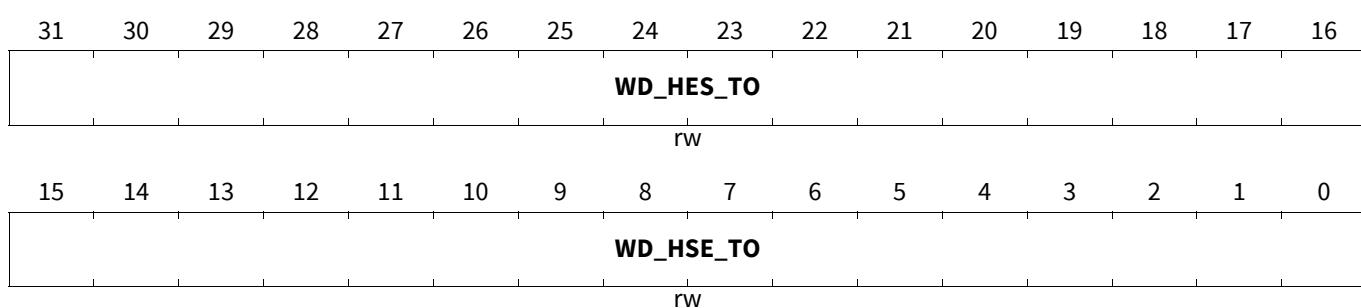
The address of each CIF Security Watchdog register is evaluated as CIF\_WATCHDOG\_BASE + Offset.

**26.4.6.1 Watchdog Configuration Registers****Watchdog Control Register****WD\_CTRL**

Field	Bits	Type	Description
<b>WD_EN</b>	0	rw	<b>Enable Security Watchdog</b> 0 <sub>B</sub> unit is disabled 1 <sub>B</sub> enables the watchdog unit
<b>RST_H_CNT</b>	1	w	<b>Reset Horizontal Counter</b> 0 <sub>B</sub> nothing happens 1 <sub>B</sub> reset counter
<b>RST_V_CNT</b>	2	w	<b>Reset Vertical Counter</b> 0 <sub>B</sub> nothing happens 1 <sub>B</sub> reset counter
<b>RST_PD_CNT</b>	3	w	<b>Reset Predivider Counter</b> 0 <sub>B</sub> nothing happens 1 <sub>B</sub> reset counter
<b>WD_PREDIV</b>	31:16	rw	<b>Watchdog Counter Predivider</b> A value of 0 means that the Watchdog Counters are increased with every CIF clock cycle. Every other value N leads to an increment at every N+1th cycle.
<b>0</b>	15:4	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****Watchdog Vertical Timeout Register****WD\_V\_TIMEOUT****Watchdog Vertical Timeout Register**(2504<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>WD_VSE_TO</b>	15:0	rw	<b>Watchdog Vertical Start End Timeout</b> A value of 0 means that the Timeout is disabled. Every other value leads to an interrupt trigger when the vertical counter reaches the value in the vertical Start End phase.
<b>WD_VES_TO</b>	31:16	rw	<b>Watchdog Vertical End Start Timeout</b> A value of 0 means that the Timeout is disabled. Every other value leads to an interrupt trigger when the vertical counter reaches the value in the vertical End Start phase.

**Watchdog Horizontal Timeout Register****WD\_H\_TIMEOUT****Watchdog Horizontal Timeout Register**(2508<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>WD_HSE_TO</b>	15:0	rw	<b>Watchdog Horizontal Start End Timeout</b> A value of 0 means that the Timeout is disabled. Every other value leads to an interrupt trigger when the horizontal counter reaches the value in the horizontal Start End phase.
<b>WD_HES_TO</b>	31:16	rw	<b>Watchdog Horizontal End Start Timeout</b> A value of 0 means that the Timeout is disabled. Every other value leads to an interrupt trigger when the horizontal counter reaches the value in the horizontal End Start phase.

## Camera and ADC Interface (CIF)

## 26.4.6.2 Watchdog Interrupt Registers

## Watchdog Interrupt Mask Register

## WD\_IMSC

Watchdog Interrupt Mask Register (250C<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												IMSC_WD_H SE_TO	IMSC_WD_H ES_TO	IMSC_WD_V SE_TO	IMSC_WD_V ES_TO
r															

Field	Bits	Type	Description
IMSC_WD_VE S_TO	0	rw	<b>Vertical End Start Timeout</b> Enable interrupt (1) or mask out (0)
IMSC_WD_VS E_TO	1	rw	<b>Vertical Start End Timeout</b> Enable interrupt (1) or mask out (0)
IMSC_WD_HE S_TO	2	rw	<b>Horizontal End Start Timeout</b> Enable interrupt (1) or mask out (0)
IMSC_WD_HS E_TO	3	rw	<b>Horizontal Start End Timeout</b> Enable interrupt (1) or mask out (0)
0	31:4	r	<b>Reserved</b> Read as 0, should be written with 0.

## Watchdog Raw Interrupt Status Register

## WD\_RIS

Watchdog Raw Interrupt Status Register (2510<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>

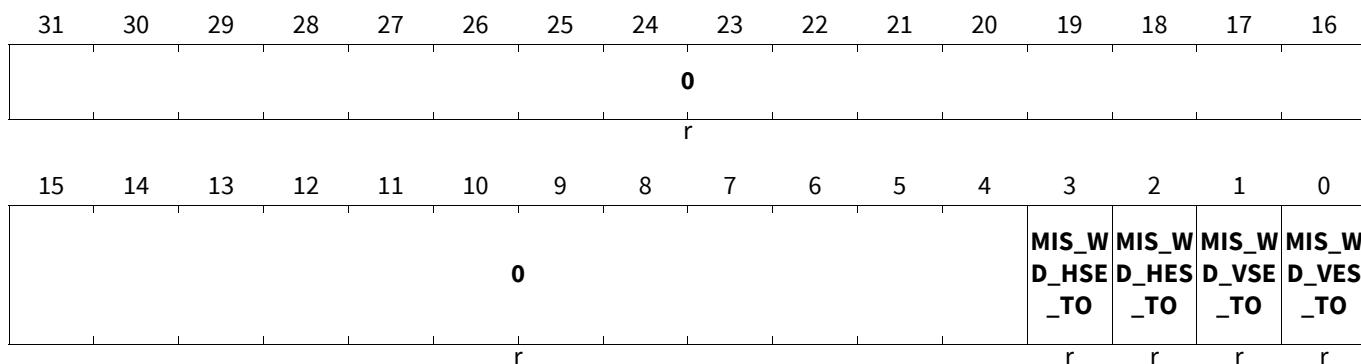
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												RIS_W D_HSE _TO	RIS_W D_HES _TO	RIS_W D_VSE _TO	RIS_W D_VES _TO
r															

## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
RIS_WD_VES_TO	0	r	<b>Vertical End Start Timeout</b> The counter reached the programmed timeout.
RIS_WD_VSE_TO	1	r	<b>Vertical Start End Timeout</b> The counter reached the programmed timeout.
RIS_WD_HES_TO	2	r	<b>Horizontal End Start Timeout</b> The counter reached the programmed timeout.
RIS_WD_HSE_TO	3	r	<b>Horizontal Start End Timeout</b> The counter reached the programmed timeout.
0	31:4	r	<b>Reserved</b> Read as 0.

## Watchdog Masked Interrupt Status Register

## WD\_MIS

Watchdog Masked Interrupt Status Register (2514<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
MIS_WD_VES_TO	0	r	<b>Vertical End Start Timeout</b> The counter reached the programmed timeout.
MIS_WD_VSE_TO	1	r	<b>Vertical Start End Timeout</b> The counter reached the programmed timeout.
MIS_WD_HES_TO	2	r	<b>Horizontal End Start Timeout</b> The counter reached the programmed timeout.
MIS_WD_HSE_TO	3	r	<b>Horizontal Start End Timeout</b> The counter reached the programmed timeout.
0	31:4	r	<b>Reserved</b> Read as 0.

## **Camera and ADC Interface (CIF)**

## Watchdog Interrupt Clear Register

WD\_ICR

## Watchdog Interrupt Clear Register

(2518)<sub>H</sub>

**Application Reset Value: 0000 0000<sub>H</sub>**

The figure consists of two horizontal bar charts. The top chart has a horizontal axis labeled 'r' with tick marks at integer intervals from 31 to 16. Above each tick mark is a numerical value: 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, and 16. A central tick mark is labeled '0'. The bottom chart has a similar horizontal axis labeled 'r' with tick marks at integer intervals from 15 to 0. Above each tick mark is a numerical value: 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, and 0. A central tick mark is labeled '0'. Both charts have a light gray background with white bars.

Field	Bits	Type	Description
<b>ICR_WD_VES_TO</b>	0	w	<b>Vertical End Start Timeout</b> 0 <sub>B</sub> nothing happens 1 <sub>B</sub> clear interrupt
<b>ICR_WD_VSE_TO</b>	1	w	<b>Vertical Start End Timeout</b> 0 <sub>B</sub> nothing happens 1 <sub>B</sub> clear interrupt
<b>ICR_WD_HES_TO</b>	2	w	<b>Horizontal End Start Timeout</b> 0 <sub>B</sub> nothing happens 1 <sub>B</sub> clear interrupt
<b>ICR_WD_HSE_TO</b>	3	w	<b>Horizontal Start End Timeout</b> 0 <sub>B</sub> nothing happens 1 <sub>B</sub> clear interrupt
<b>0</b>	31:4	r	<b>Reserved</b> Read as 0, should be written with 0.

## Watchdog Interrupt Set Register

WD ISR

## Watchdog Interrupt Set Register

(251C<sub>4</sub>)

**Application Reset Value: 0000 0000,**

## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
<b>ISR_WD_VES_TO</b>	0	w	<b>Vertical End Start Timeout</b> 0 <sub>B</sub> nothing happens 1 <sub>B</sub> set interrupt
<b>ISR_WD_VSE_TO</b>	1	w	<b>Vertical Start End Timeout</b> 0 <sub>B</sub> nothing happens 1 <sub>B</sub> set interrupt
<b>ISR_WD_HES_TO</b>	2	w	<b>Horizontal End Start Timeout</b> 0 <sub>B</sub> nothing happens 1 <sub>B</sub> set interrupt
<b>ISR_WD_HSE_TO</b>	3	w	<b>Horizontal Start End Timeout</b> 0 <sub>B</sub> nothing happens 1 <sub>B</sub> set interrupt
<b>0</b>	31:4	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****26.4.7 ISP Image Stabilization Registers**

The address of each CIF image stabilization register is evaluated as CIF\_IS\_BASE + Offset.

**26.4.7.1 Image Stabilization Control Registers****ISP Image Stabilization Control Register****ISPIS\_CTRL**

**ISP Image Stabilization Control Register (2400<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							0								
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							0								
r															rw

Field	Bits	Type	Description
IS_EN	0	rw	<b>Image Stabilization Enable</b> 0 <sub>B</sub> image stabilization switched off 1 <sub>B</sub> image stabilization switched on
0	31:1	r	<b>Reserved</b> Read as 0, should be written with 0.

**ISP Image Stabilization Recenter Register****ISPIS\_RECENTER**

**ISP Image Stabilization Recenter Register (2404<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							0								
r															

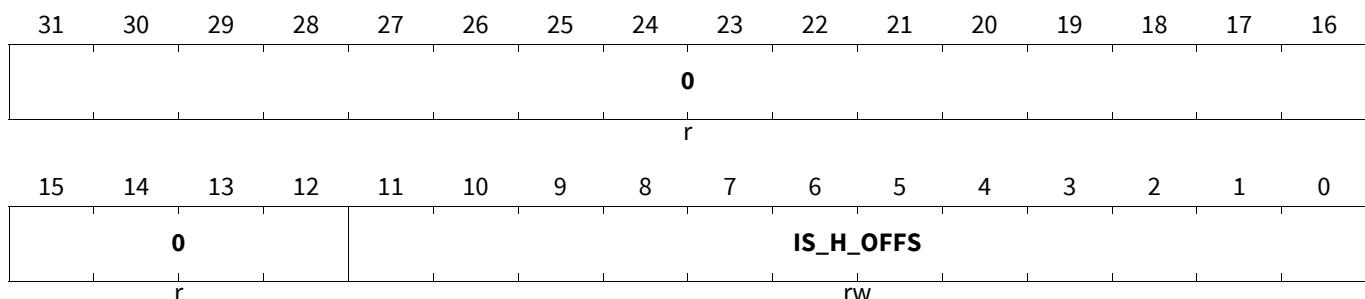
  

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							0								
r															rw

Field	Bits	Type	Description
RECENTER	2:0	rw	<b>Recenter</b> For all other values recentering is active (cur_h/v_offs-H/V_OFFSETS)/2 000 <sub>B</sub> recenter feature switched off
0	31:3	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****ISP Image Stabilization Horizontal Offset Of Output Window Register****ISPIIS\_H\_OFFSETS**

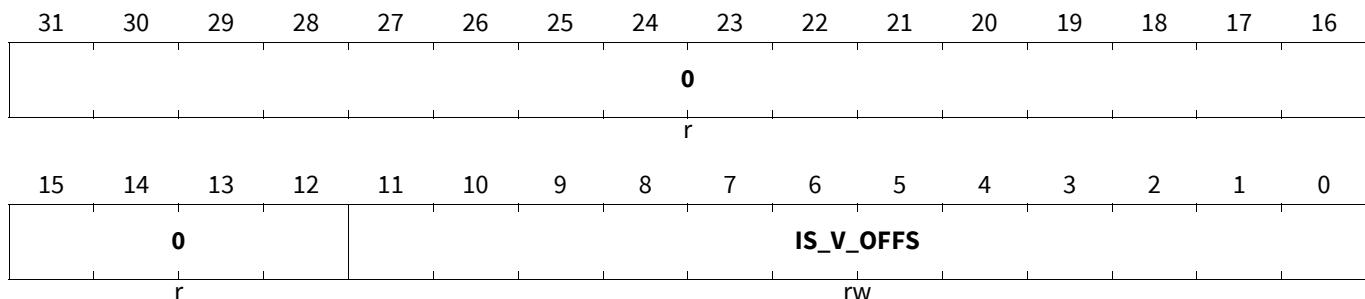
**ISP Image Stabilization Horizontal Offset Of Output Window Register(2408<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**



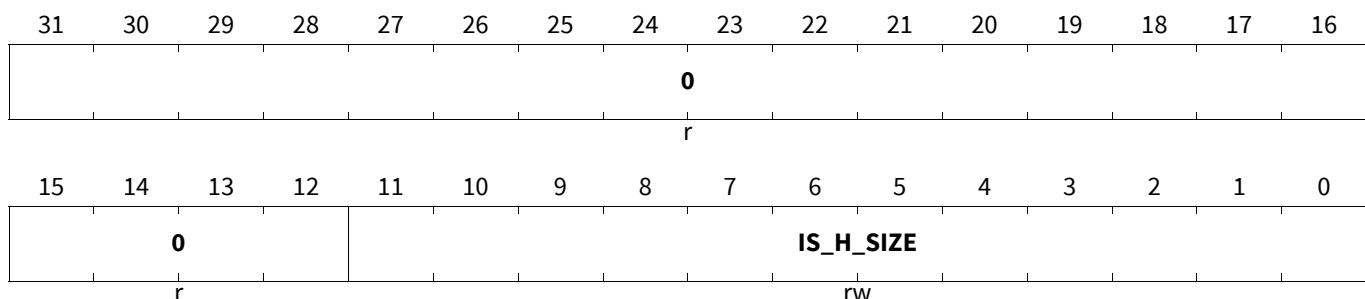
Field	Bits	Type	Description
<b>IS_H_OFFSETS</b>	11:0	rw	<b>Horizontal Picture Offset</b> Horizontal picture offset in pixel
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**ISP Image Stabilization Vertical Offset Of Output Window Register****ISPIIS\_V\_OFFSETS**

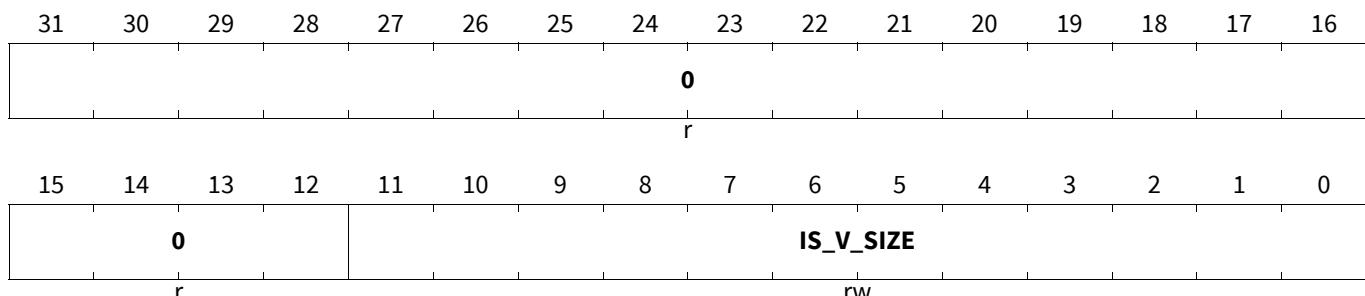
**ISP Image Stabilization Vertical Offset Of Output Window Register(240C<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>IS_V_OFFSETS</b>	11:0	rw	<b>Vertical Picture Offset</b> Vertical picture offset in lines
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****ISP Image Stabilization Output Horizontal Picture Size Register****ISPIIS\_H\_SIZE****ISP Image Stabilization Output Horizontal Picture Size Register(2410<sub>H</sub>)****0A28<sub>H</sub>****Application Reset Value: 0000**

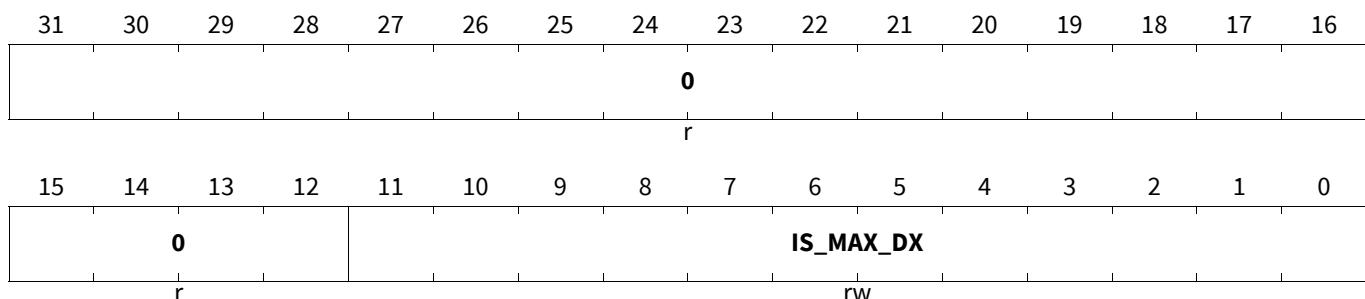
Field	Bits	Type	Description
<b>IS_H_SIZE</b>	11:0	rw	<b>Horizontal Picture Size</b> Horizontal picture size in pixel Only even numbers are accepted, because complete quadruples of YUYV(YCbYCr) are needed for the following modules. If an odd size is programmed the value will be truncated to even size. If ISP_MODE is set to 001 <sub>H</sub> (ITU-R BT.656 YUV) 002 <sub>H</sub> (ITU-R BT.601 YUV) 003 <sub>H</sub> (ITU-R BT.601 Bayer RGB) 005 <sub>H</sub> (ITU-R BT.656 Bayer RGB)
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**ISP Image Stabilization Output Vertical Picture Size Register****ISPIIS\_V\_SIZE****ISP Image Stabilization Output Vertical Picture Size Register(2414<sub>H</sub>) Application Reset Value: 0000 0800<sub>H</sub>**

Field	Bits	Type	Description
<b>IS_V_SIZE</b>	11:0	rw	<b>Vertical Picture Size</b> Vertical picture size in lines
<b>0</b>	31:12	r	<b>Reserved</b>

**Camera and ADC Interface (CIF)****ISP Image Stabilization Maximum Horizontal Displacement Register****ISPIIS\_MAX\_DX**

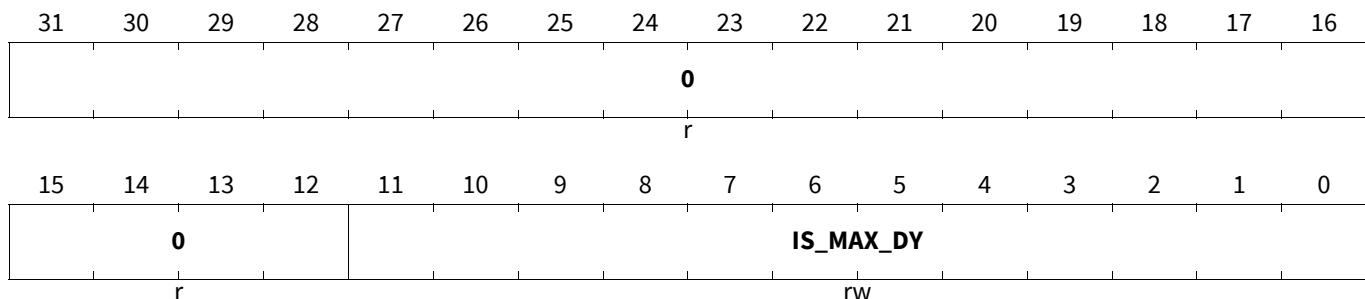
**ISP Image Stabilization Maximum Horizontal Displacement Register( $2418_H$ ) Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>IS_MAX_DX</b>	11:0	rw	<b>Maximum Horizontal Displacement</b> Maximum allowed accumulated horizontal displacement in pixels
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**ISP Image Stabilization Maximum Vertical Displacement Register****ISPIIS\_MAX\_DY**

**ISP Image Stabilization Maximum Vertical Displacement Register( $241C_H$ ) Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>IS_MAX_DY</b>	11:0	rw	<b>Maximum Vertical Displacement</b> Maximum allowed accumulated vertical displacement in lines
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

## Camera and ADC Interface (CIF)

### ISP Image Stabilization Camera Displacement Register

#### ISPIIS\_DISPLACE

ISP Image Stabilization Camera Displacement Register( $2420_H$ )

Application Reset Value:  $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				0	<b>DY</b>										
r					rw										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				0	<b>DX</b>										
r					rw										

Field	Bits	Type	Description
<b>DX</b>	11:0	rw	<b>Camera Displacement</b> ISP_IS will compensate for horizontal camera displacement of DX pixels in the next frame
<b>DY</b>	27:16	rw	<b>Camera Displacement</b> ISP_IS will compensate for vertical camera displacement of DY pixels in the next frame
<b>0</b>	15:12, 31:28	r	<b>Reserved</b> Read as 0, should be written with 0.

### 26.4.7.2 Image Stabilization Shadow Registers

#### ISP Image Current Horizontal Offset Of Output Window Shadow Register

#### ISPIIS\_H\_OFFSET\_SHD

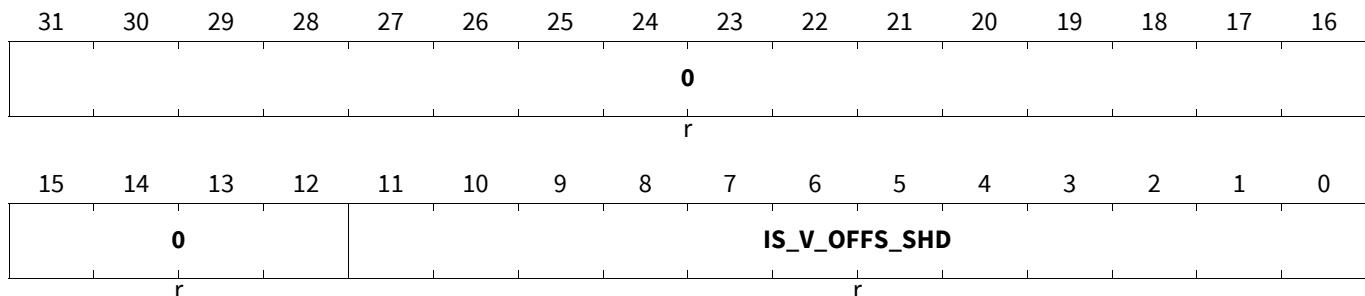
ISP Image Current Horizontal Offset Of Output Window Shadow Register( $2424_H$ ) Application Reset Value:  $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
					0	<b>0</b>									
r						r									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				0	<b>IS_H_OFFSET_SHD</b>										
r					r										

Field	Bits	Type	Description
<b>IS_H_OFFSET_S</b>	12:0	r	<b>Horizontal Picture Offset</b> Current horizontal picture offset in lines
<b>0</b>	31:13	r	<b>Reserved</b> Read as 0

**Camera and ADC Interface (CIF)****ISP Image Current Vertical Offset Of Output Window Shadow Register****ISPIS\_V\_OFFSET\_SHD**

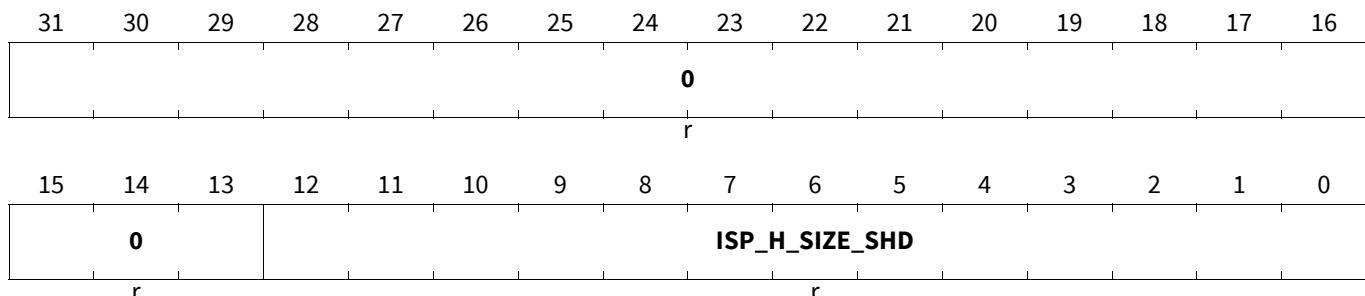
**ISP Image Current Vertical Offset Of Output Window Shadow Register(2428<sub>H</sub>)** Application Reset Value:  
0000 0000<sub>H</sub>



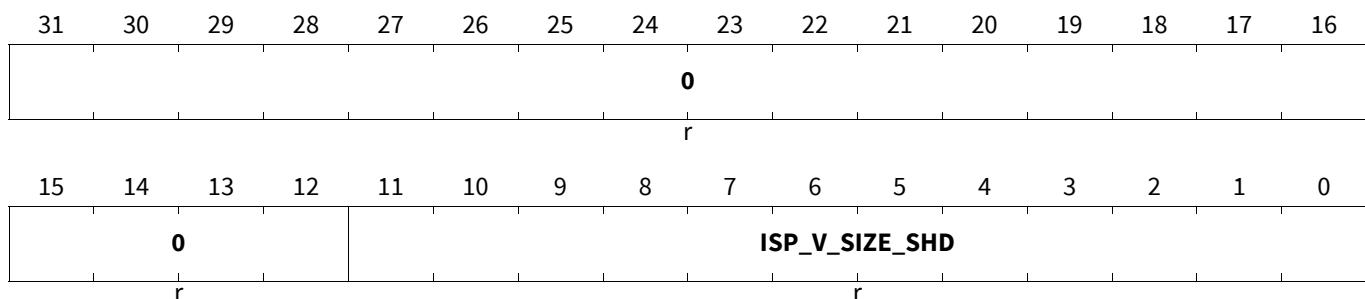
Field	Bits	Type	Description
<b>IS_V_OFFSET_SHD</b>	11:0	r	<b>Vertical Picture Offset</b> Current vertical picture offset in lines
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0.

**ISP Image Current Output Horizontal Picture Size Shadow Register****ISPIS\_H\_SIZE\_SHD**

**ISP Image Current Output Horizontal Picture Size Shadow Register(242C<sub>H</sub>)** Application Reset Value: 0000 0000<sub>H</sub>



Field	Bits	Type	Description
<b>ISP_H_SIZE_SHD</b>	12:0	r	<b>Horizontal Picture Size</b> Current horizontal picture size in pixel
<b>0</b>	31:13	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****ISP Image Current Output Vertical Picture Size Shadow Register****ISPIS\_V\_SIZE\_SHD****ISP Image Current Output Vertical Picture Size Shadow Register(2430<sub>H</sub>)****Application Reset Value: 0000****0000<sub>H</sub>**

Field	Bits	Type	Description
<b>ISP_V_SIZE_SHD</b>	11:0	r	<b>Vertical Picture Size</b> Current vertical picture size in lines
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0.

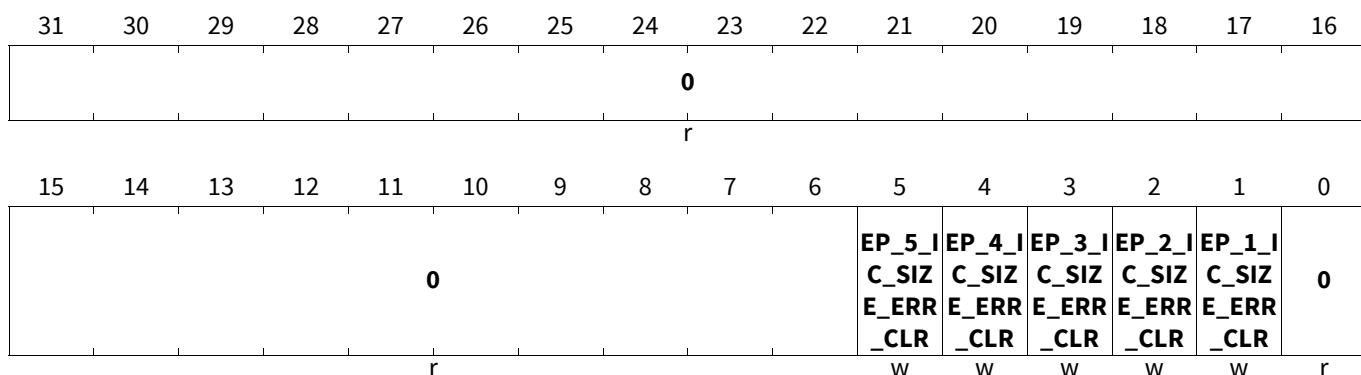
**Camera and ADC Interface (CIF)****26.4.8 Extra Path Programming Registers**

The address of each CIF Extra Path register is evaluated as CIF\_EXTRA\_PATH\_BASE + Offset.

**26.4.8.1 Extra Path Error Registers****Extra Path Error Register****MIEP\_STA\_ERR****Extra Path Error Register**(3500<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0											EP_5_FIFO_FULL	EP_4_FIFO_FULL	EP_3_FIFO_FULL	EP_2_FIFO_FULL	EP_1_FIFO_FULL	0
r										r	r	r	r	r	r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0											EP_5_I_C_SIZ_E_ERR	EP_4_I_C_SIZ_E_ERR	EP_3_I_C_SIZ_E_ERR	EP_2_I_C_SIZ_E_ERR	EP_1_I_C_SIZ_E_ERR	0
r										r	r	r	r	r	r	

Field	Bits	Type	Description
<b>EP_1_IC_SIZE_ERR</b>	1	r	<b>Size error is generated in Extra Path 1 image cropping submodule</b>
<b>EP_2_IC_SIZE_ERR</b>	2	r	<b>Size error is generated in Extra Path 2 image cropping submodule</b>
<b>EP_3_IC_SIZE_ERR</b>	3	r	<b>Size error is generated in Extra Path 3 image cropping submodule</b>
<b>EP_4_IC_SIZE_ERR</b>	4	r	<b>Size error is generated in Extra Path 4 image cropping submodule</b>
<b>EP_5_IC_SIZE_ERR</b>	5	r	<b>Size error is generated in Extra Path 5 image cropping submodule</b>
<b>EP_1_FIFO_FULL</b>	17	r	<b>Extra Path 1 FIFO Full</b> FIFO full flag of FIFO in extra path asserted since last clear
<b>EP_2_FIFO_FULL</b>	18	r	<b>Extra Path 2 FIFO Full</b> FIFO full flag of FIFO in extra path asserted since last clear
<b>EP_3_FIFO_FULL</b>	19	r	<b>Extra Path 3 FIFO Full</b> FIFO full flag of FIFO in extra path asserted since last clear
<b>EP_4_FIFO_FULL</b>	20	r	<b>Extra Path 4 FIFO Full</b> FIFO full flag of FIFO in extra path asserted since last clear
<b>EP_5_FIFO_FULL</b>	21	r	<b>Extra Path 5 FIFO Full</b> FIFO full flag of FIFO in extra path asserted since last clear
<b>0</b>	0, 16:6, 31:22	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****Extra Path Status Error Clear Register****MIEP\_STA\_ERR\_CLR****Extra Path Status Error Clear Register**(3504<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
EP_1_IC_SIZE _ERR_CLR	1	w	<b>Size error is cleared</b>
EP_2_IC_SIZE _ERR_CLR	2	w	<b>Size error is cleared</b>
EP_3_IC_SIZE _ERR_CLR	3	w	<b>Size error is cleared</b>
EP_4_IC_SIZE _ERR_CLR	4	w	<b>Size error is cleared</b>
EP_5_IC_SIZE _ERR_CLR	5	w	<b>Size error is cleared</b>
0	0, 31:6	r	<b>Reserved</b> Read as 0, should be written with 0.

## Camera and ADC Interface (CIF)

## 26.4.8.2 Memory Interface Extra Path Interrupt Registers

MI Extra Path Interrupt Mask '1': interrupt active, '0': interrupt masked

## MIEP\_IMSC

MI Extra Path Interrupt Mask '1': interrupt active, '0': interrupt masked(3508<sub>H</sub>) Application Reset Value:  
0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBLK _LINE _EP_4	WRAP _EP_4	FILL_E P_4	FRAM E-END _EP_4	MBLK _LINE _EP_3	WRAP _EP_3	FILL_E P_3	FRAM E-END _EP_3	MBLK _LINE _EP_2	WRAP _EP_2	FILL_E P_2	FRAM E-END _EP_2	MBLK _LINE _EP_1	WRAP _EP_1	FILL_E P_1	FRAM E-END _EP_1
rw	rw	rw	rw												

Field	Bits	Type	Description
FRAME_END_EP_1	0	rw	<b>Extra Path 1 Frame End</b> Mask Extra Path end of frame interrupt
FILL_EP_1	1	rw	<b>Fill Extra Path 1</b> Mask bit for fill level interrupt of Extra Path
WRAP_EP_1	2	rw	<b>Wrap Extra Path 1</b> Mask bit for Extra Path address wrap interrupt
MBLK_LINE_E_P_1	3	rw	<b>Macro Block Line Interrupt Extra Path 1</b> Mask bit for macroblock line interrupt of Extra Path 1 (16 lines are written into RAM)
FRAME_END_EP_2	4	rw	<b>Extra Path 2 Frame End</b> Mask Extra Path end of frame interrupt
FILL_EP_2	5	rw	<b>Fill Extra Path 2</b> Mask bit for fill level interrupt of Extra Path
WRAP_EP_2	6	rw	<b>Wrap Extra Path 2</b> Mask bit for Extra Path address wrap interrupt
MBLK_LINE_E_P_2	7	rw	<b>Macro Block Line Interrupt Extra Path 2</b> Mask bit for macroblock line interrupt of Extra Path 2 (16 lines are written into RAM)
FRAME_END_EP_3	8	rw	<b>Extra Path 3 Frame End</b> Mask Extra Path end of frame interrupt
FILL_EP_3	9	rw	<b>Fill Extra Path 3</b> Mask bit for fill level interrupt of Extra Path
WRAP_EP_3	10	rw	<b>Wrap Extra Path 3</b> Mask bit for Extra Path address wrap interrupt

## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
<b>MBLK_LINE_E_P_3</b>	11	rw	<b>Macro Block Line Interrupt Extra Path 3</b> Mask bit for macroblock line interrupt of Extra Path 3 (16 lines are written into RAM)
<b>FRAME_END_EP_4</b>	12	rw	<b>Extra Path 4 Frame End</b> Mask Extra Path end of frame interrupt
<b>FILL_EP_4</b>	13	rw	<b>Fill Extra Path 4</b> Mask bit for fill level interrupt of Extra Path
<b>WRAP_EP_4</b>	14	rw	<b>Wrap Extra Path 4</b> Mask bit for Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_4</b>	15	rw	<b>Macro Block Line Interrupt Extra Path 4</b> Mask bit for macroblock line interrupt of Extra Path 4 (16 lines are written into RAM)
<b>FRAME_END_EP_5</b>	16	rw	<b>Extra Path 5 Frame End</b> Mask Extra Path end of frame interrupt
<b>FILL_EP_5</b>	17	rw	<b>Fill Extra Path 5</b> Mask bit for fill level interrupt of Extra Path
<b>WRAP_EP_5</b>	18	rw	<b>Wrap Extra Path 5</b> Mask bit for Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_5</b>	19	rw	<b>Macro Block Line Interrupt Extra Path 5</b> Mask bit for macroblock line interrupt of Extra Path 5 (16 lines are written into RAM)
<b>0</b>	31:20	r	<b>Reserved</b> Read as 0, should be written with 0.

## MI Extra Path Raw Interrupt Status Register

## MIEP\_RIS

MI Extra Path Raw Interrupt Status Register (350C<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
<b>MBLK_LINE_EP_4</b>	<b>WRAP_EP_4</b>	<b>FILL_E_P_4</b>	<b>FRAM_E_END_EP_4</b>	<b>MBLK_LINE_EP_3</b>	<b>WRAP_EP_3</b>	<b>FILL_E_P_3</b>	<b>FRAM_E_END_EP_3</b>	<b>MBLK_LINE_EP_2</b>	<b>WRAP_EP_2</b>	<b>FILL_E_P_2</b>	<b>FRAM_E_END_EP_2</b>	<b>MBLK_LINE_EP_1</b>	<b>WRAP_EP_1</b>	<b>FILL_E_P_1</b>	<b>FRAM_E_END_EP_1</b>
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
<b>FRAME_END_EP_1</b>	0	r	<b>Extra Path 1 Frame End</b> Raw status Extra Path end of frame interrupt

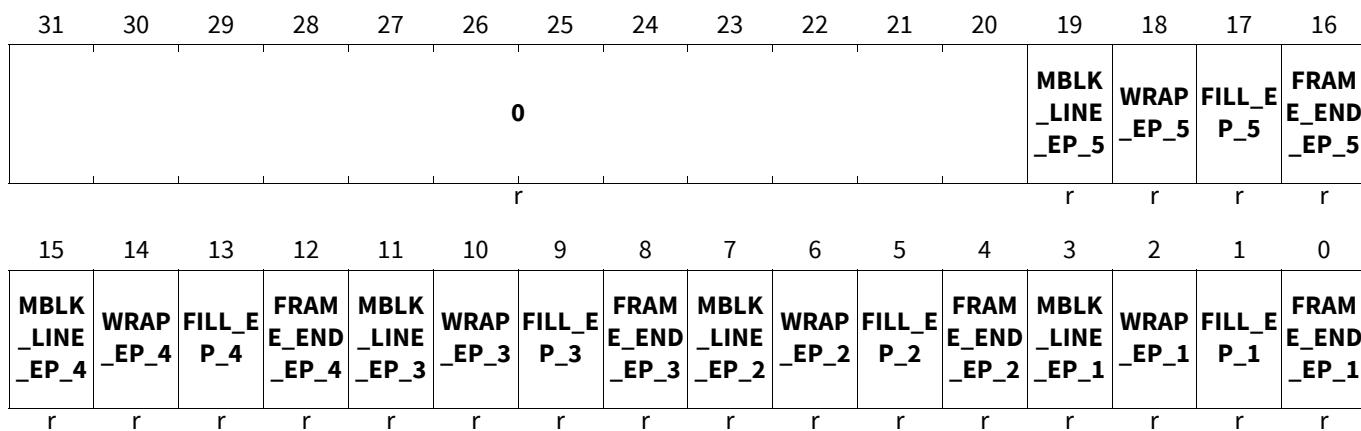
## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
<b>FILL_EP_1</b>	1	r	<b>Fill Extra Path 1</b> Raw status for fill level interrupt of Extra Path
<b>WRAP_EP_1</b>	2	r	<b>Wrap Extra Path 1</b> Raw status for Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_1</b>	3	r	<b>Macro Block Line Interrupt Extra Path 1</b> Raw status for macroblock line interrupt of Extra Path 1 (16 lines are written into RAM)
<b>FRAME_END_EP_2</b>	4	r	<b>Extra Path 2 Frame End</b> Raw status Extra Path end of frame interrupt
<b>FILL_EP_2</b>	5	r	<b>Fill Extra Path 2</b> Raw status for fill level interrupt of Extra Path
<b>WRAP_EP_2</b>	6	r	<b>Wrap Extra Path 2</b> Raw status for Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_2</b>	7	r	<b>Macro Block Line Interrupt Extra Path 2</b> Raw status for macroblock line interrupt of Extra Path 2 (16 lines are written into RAM)
<b>FRAME_END_EP_3</b>	8	r	<b>Extra Path 3 Frame End</b> Raw status Extra Path end of frame interrupt
<b>FILL_EP_3</b>	9	r	<b>Fill Extra Path 3</b> Raw status for fill level interrupt of Extra Path
<b>WRAP_EP_3</b>	10	r	<b>Wrap Extra Path 3</b> Raw status for Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_3</b>	11	r	<b>Macro Block Line Interrupt Extra Path 3</b> Raw status for macroblock line interrupt of Extra Path 3 (16 lines are written into RAM)
<b>FRAME_END_EP_4</b>	12	r	<b>Extra Path 4 Frame End</b> Raw status Extra Path end of frame interrupt
<b>FILL_EP_4</b>	13	r	<b>Fill Extra Path 4</b> Raw status for fill level interrupt of Extra Path
<b>WRAP_EP_4</b>	14	r	<b>Wrap Extra Path 4</b> Raw status for Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_4</b>	15	r	<b>Macro Block Line Interrupt Extra Path 4</b> Raw status for macroblock line interrupt of Extra Path 4 (16 lines are written into RAM)
<b>FRAME_END_EP_5</b>	16	r	<b>Extra Path 5 Frame End</b> Raw status picture end of frame interrupt
<b>FILL_EP_5</b>	17	r	<b>Fill Extra Path 5</b> Raw status for fill level interrupt of Extra Path
<b>WRAP_EP_5</b>	18	r	<b>Wrap Extra Path 5</b> Raw status for Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_5</b>	19	r	<b>Macro Block Line Interrupt Extra Path 5</b> Raw status for macroblock line interrupt of Extra Path 5 (16 lines are written into RAM)

## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
0	31:20	r	<b>Reserved</b> Read as 0.

## MI Extra Path Masked Interrupt Status Register

**MIEP\_MIS****MI Extra Path Masked Interrupt Status Register (3510<sub>H</sub>)**Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
FRAME_END_EP_1	0	r	<b>Extra Path 1 Frame End</b> Masked status Extra Path end of frame interrupt
FILL_EP_1	1	r	<b>Fill Extra Path 1</b> Masked status for fill level interrupt of Extra Path
WRAP_EP_1	2	r	<b>Wrap Extra Path 1</b> Masked status for Extra Path address wrap interrupt
MBLK_LINE_E_P_1	3	r	<b>Macro Block Line Interrupt Extra Path 1</b> Masked status for macroblock line interrupt of Extra Path 1 (16 lines are written into RAM)
FRAME_END_EP_2	4	r	<b>Extra Path 2 Frame End</b> Masked status Extra Path end of frame interrupt
FILL_EP_2	5	r	<b>Fill Extra Path 2</b> Masked status for fill level interrupt of Extra Path
WRAP_EP_2	6	r	<b>Wrap Extra Path 2</b> Masked status for Extra Path address wrap interrupt
MBLK_LINE_E_P_2	7	r	<b>Macro Block Line Interrupt Extra Path 2</b> Masked status for macroblock line interrupt of Extra Path 2 (16 lines are written into RAM)
FRAME_END_EP_3	8	r	<b>Extra Path 3 Frame End</b> Masked status Extra Path end of frame interrupt
FILL_EP_3	9	r	<b>Fill Extra Path 3</b> Masked status for fill level interrupt of Extra Path
WRAP_EP_3	10	r	<b>Wrap Extra Path 3</b> Masked status for Extra Path address wrap interrupt

## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
<b>MBLK_LINE_E_P_3</b>	11	r	<b>Macro Block Line Interrupt Extra Path 3</b> Masked status for macroblock line interrupt of Extra Path 3 (16 lines are written into RAM)
<b>FRAME_END_EP_4</b>	12	r	<b>Extra Path 4 Frame End</b> Masked status Extra Path end of frame interrupt
<b>FILL_EP_4</b>	13	r	<b>Fill Extra Path 4</b> Masked status for fill level interrupt of Extra Path
<b>WRAP_EP_4</b>	14	r	<b>Wrap Extra Path 4</b> Masked status for Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_4</b>	15	r	<b>Macro Block Line Interrupt Extra Path 4</b> Masked status for macroblock line interrupt of Extra Path 4 (16 lines are written into RAM)
<b>FRAME_END_EP_5</b>	16	r	<b>Extra Path 5 Frame End</b> Masked status picture end of frame interrupt
<b>FILL_EP_5</b>	17	r	<b>Fill Extra Path 5</b> Masked status for fill level interrupt of Extra Path
<b>WRAP_EP_5</b>	18	r	<b>Wrap Extra Path 5</b> Masked status for Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_5</b>	19	r	<b>Macro Block Line Interrupt Extra Path 5</b> Masked status for macroblock line interrupt of Extra Path 5 (16 lines are written into RAM)
<b>0</b>	31:20	r	<b>Reserved</b> Read as 0.

## MI Extra Path Interrupt Clear Register

## MIEP\_ICR

## MI Extra Path Interrupt Clear Register

(3514<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
												<b>MBLK_LINE_EP_5</b>	<b>WRAP_EP_5</b>	<b>FILL_E_P_5</b>	<b>FRAM_E_END_EP_5</b>
												r	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>MBLK_LINE_EP_4</b>	<b>WRAP_EP_4</b>	<b>FILL_E_P_4</b>	<b>FRAM_E_END_EP_4</b>	<b>MBLK_LINE_EP_3</b>	<b>WRAP_EP_3</b>	<b>FILL_E_P_3</b>	<b>FRAM_E_END_EP_3</b>	<b>0</b>	<b>WRAP_EP_2</b>	<b>FILL_E_P_2</b>	<b>FRAM_E_END_EP_2</b>	<b>MBLK_LINE_EP_1</b>	<b>WRAP_EP_1</b>	<b>FILL_E_P_1</b>	<b>FRAM_E_END_EP_1</b>
w	w	w	w	w	w	w	w	r	w	w	w	w	w	w	w

Field	Bits	Type	Description
<b>FRAME_END_EP_1</b>	0	w	<b>Extra Path 1 Frame End</b> Clear Extra Path end of frame interrupt

## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
<b>FILL_EP_1</b>	1	w	<b>Fill Extra Path 1</b> Clear fill level interrupt of Extra Path
<b>WRAP_EP_1</b>	2	w	<b>Wrap Extra Path 1</b> Clear Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_1</b>	3	w	<b>Macro Block Line Interrupt Extra Path 1</b> Clear macroblock line interrupt of Extra Path 1
<b>FRAME_END_EP_2</b>	4	w	<b>Extra Path 2 Frame End</b> Clear Extra Path end of frame interrupt
<b>FILL_EP_2</b>	5	w	<b>Fill Extra Path 2</b> Clear fill level interrupt of Extra Path
<b>WRAP_EP_2</b>	6	w	<b>Wrap Extra Path 2</b> Clear Extra Path address wrap interrupt
<b>FRAME_END_EP_3</b>	8	w	<b>Extra Path 3 Frame End</b> Clear Extra Path end of frame interrupt
<b>FILL_EP_3</b>	9	w	<b>Fill Extra Path 3</b> Clear fill level interrupt of Extra Path
<b>WRAP_EP_3</b>	10	w	<b>Wrap Extra Path 3</b> Clear Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_3</b>	11	w	<b>Macro Block Line Interrupt Extra Path 3</b> Clear macroblock line interrupt of Extra Path 3
<b>FRAME_END_EP_4</b>	12	w	<b>Extra Path 4 Frame End</b> Clear Extra Path end of frame interrupt
<b>FILL_EP_4</b>	13	w	<b>Fill Extra Path 4</b> Clear fill level interrupt of Extra Path
<b>WRAP_EP_4</b>	14	w	<b>Wrap Extra Path 4</b> Clear Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_4</b>	15	w	<b>Macro Block Line Interrupt Extra Path 4</b> Clear macroblock line interrupt of Extra Path 4
<b>FRAME_END_EP_5</b>	16	w	<b>Extra Path 5 Frame End</b> Clear picture end of frame interrupt
<b>FILL_EP_5</b>	17	w	<b>Fill Extra Path 5</b> Clear fill level interrupt of Extra Path
<b>WRAP_EP_5</b>	18	w	<b>Wrap Extra Path 5</b> Clear Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_5</b>	19	w	<b>Macro Block Line Interrupt Extra Path 5</b> Clear macroblock line interrupt of Extra Path 5
<b>0</b>	7, 31:20	r	<b>Reserved</b> Read as 0, should be written with 0.

## Camera and ADC Interface (CIF)

## MI Extra Path Interrupt Set Register

## MIEP\_ISR

## MI Extra Path Interrupt Set Register

(3518<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBLK_LINE_EP_4	WRAP_EP_4	FILL_E_P_4	FRAM_E_END_EP_4	MBLK_LINE_EP_3	WRAP_EP_3	FILL_E_P_3	FRAM_E_END_EP_3	MBLK_LINE_EP_2	WRAP_EP_2	FILL_E_P_2	FRAM_E_END_EP_2	MBLK_LINE_EP_1	WRAP_EP_1	FILL_E_P_1	FRAM_E_END_EP_1
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Field	Bits	Type	Description
FRAME-END_EP_1	0	w	<b>Extra Path 1 Frame End</b> Set Extra Path end of frame interrupt
FILL_EP_1	1	w	<b>Fill Extra Path 1</b> Set fill level interrupt of Extra Path
WRAP_EP_1	2	w	<b>Wrap Extra Path 1</b> Set Extra Path address wrap interrupt
MBLK_LINE_E_P_1	3	w	<b>Macro Block Line Interrupt Extra Path 1</b> Set macroblock line interrupt of Extra Path 1
FRAME-END_EP_2	4	w	<b>Extra Path 2 Frame End</b> Set Extra Path end of frame interrupt
FILL_EP_2	5	w	<b>Fill Extra Path 2</b> Set fill level interrupt of Extra Path
WRAP_EP_2	6	w	<b>Wrap Extra Path 2</b> Set Extra Path address wrap interrupt
MBLK_LINE_E_P_2	7	w	<b>Macro Block Line Interrupt Extra Path 2</b> Set macroblock line interrupt of Extra Path 2
FRAME-END_EP_3	8	w	<b>Extra Path 3 Frame End</b> Set Extra Path end of frame interrupt
FILL_EP_3	9	w	<b>Fill Extra Path 3</b> Set fill level interrupt of Extra Path
WRAP_EP_3	10	w	<b>Wrap Extra Path 3</b> Set Extra Path address wrap interrupt
MBLK_LINE_E_P_3	11	w	<b>Macro Block Line Interrupt Extra Path 3</b> Set macroblock line interrupt of Extra Path 3
FRAME-END_EP_4	12	w	<b>Extra Path 4 Frame End</b> Set Extra Path end of frame interrupt
FILL_EP_4	13	w	<b>Fill Extra Path 4</b> Set fill level interrupt of Extra Path

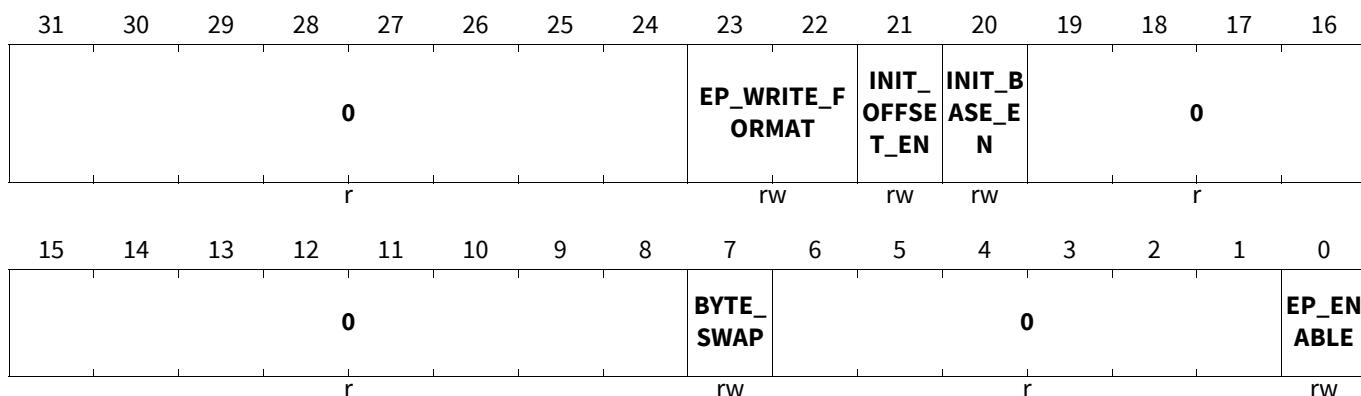
## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
<b>WRAP_EP_4</b>	14	w	<b>Wrap Extra Path 4</b> Set Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_4</b>	15	w	<b>Macro Block Line Interrupt Extra Path 4</b> Set macroblock line interrupt of Extra Path 4
<b>FRAME_END_EP_5</b>	16	w	<b>Extra Path 5 Frame End</b> Set picture end of frame interrupt
<b>FILL_EP_5</b>	17	w	<b>Fill Extra Path 5</b> Set fill level interrupt of Extra Path
<b>WRAP_EP_5</b>	18	w	<b>Wrap Extra Path 5</b> Set Extra Path address wrap interrupt
<b>MBLK_LINE_E_P_5</b>	19	w	<b>Macro Block Line Interrupt Extra Path 5</b> Set macroblock line interrupt of Extra Path 5
<b>0</b>	31:20	r	<b>Reserved</b> Read as 0, should be written with 0.

## 26.4.8.3 Memory Interface Extra Path Control Registers

## Memory Interface Extra Path j Control Register

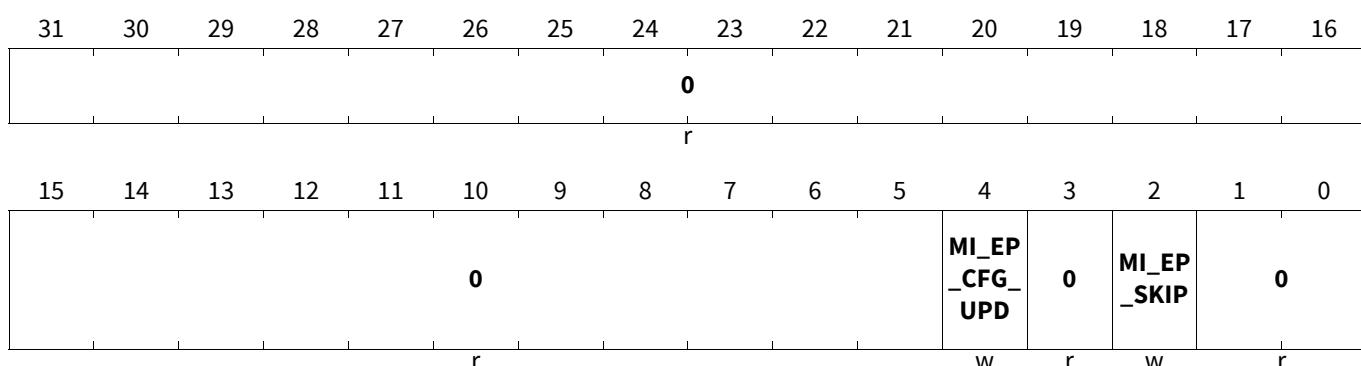
## MIEP\_j\_CTRL (j=0-4)

Memory Interface Extra Path j Control Register( $3600_H+j*100_H$ )Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>EP_ENABLE</b>	0	rw	<b>Enables enable ep picture data path</b> Programmed value becomes effective (visible in control shadow register) after a soft reset, a forced software update or an automatic config update. Affects MI_IN and MI_OUT module.

## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
BYTE_SWAP	7	rw	<p><b>Byte Swap Enable</b>            Enables change of byte order of the 32 bit output word at write port.</p> <p><i>Note:</i> Programmed value becomes effective immediately. So write to the register only if no picture data is sent.</p> <p><math>0_B</math> no byte mirroring  <math>1_B</math> byte order is mirrored but the bit order within one byte doesn't change</p>
INIT_BASE_EN	20	rw	<p><b>Init Base Address Enable</b>            Enables updating of the base address and buffer size shadow registers for picture to the programmed register init values.</p> <p>MI_EP_BASE_AD_INIT            -&gt; MI_EP_BASE_AD_SHD            MI_EP_SIZE_INIT            -&gt; MI_EP_SIZE_SHD</p> <p>The update will be executed either when a forced software update occurs (in register MI_EP_n_INIT bit cfg_upd = 1) or when an automatic config update signal arrives at the MI input port. So only the corresponding shadow registers are affected.</p>
INIT_OFFSET_EN	21	rw	<p><b>Init Offset Counter Enable</b>            Enables updating of the offset counters shadow registers for Extra Path to the programmed register init values.</p> <p>MI_EP_OFFS_CNT_INIT            -&gt; MI_EP_OFFS_CNT_SHD</p> <p>The update will be executed either when a forced software update occurs (in register MI_EP_n_INIT bit cfg_upd = 1) or when an automatic config update signal arrives at the MI input port. So only the corresponding shadow registers are affected.</p> <p>After a picture skip has been performed init_offset_en selects between skip restart and skip init mode (see bit skip in register MI_EP_INIT).</p>
EP_WRITE_FORMAT	23:22	rw	<p><b>Extra Path Write Format</b>            Defines how Extra Path data is written to memory.</p> <p><i>Note:</i> Programmed value becomes effective immediately. So write to the register only if no picture data is sent to the path.</p> <p><math>00_B</math> RAW &amp; data mode (8 bit)  <math>01_B</math> RAW 8 bit  <math>10_B</math> RAW &amp; data mode (greater 8 up to 16 bit)  <math>11_B</math> YCbCr 16 bit; YCbCr data is handled interleaved as 16 bit data in extra paths.</p>
0	6:1, 19:8, 31:24	r	<p><b>Reserved</b>            Read as 0, should be written with 0.</p>

**Camera and ADC Interface (CIF)****Memory Interface Extra Path j Control Register For Address Init And Skip Function Register****MIEP\_j\_INIT (j=0-4)****Memory Interface Extra Path j Control Register For Address Init And Skip Function Register(3604<sub>H</sub>+j\*100<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>MI_EP_SKIP</b>	2	w	<p><b>Skip Picture</b>  Skip of current or next starting main picture:  Aborts writing of main picture image data of the current frame to RAM  (after the current burst transmission has been completed). Further  picture data up to the end of the current frame are discarded.No further  macroblock line interrupt (mblk_line), no wrap around interrupt for  picture (wrap_ep_n) and no fill level interrupt (fill_ep_n) are  generated.Skip does not affect the generation of the extra path frame  end interrupt (ep_frame_end).The byte counter (register  MI_EP_BYTE_CNT) is not affected. It produces the correct number of  RAW data bytes at the end of the current (skipped) frame.After a skip has  been performed the offset counter for the main picture at the start of the  following frame are set depending on the bit init_offset_en in register  MI_CTRL:</p> <p><b>A)</b> Skip restart mode (init_offset_en = 0)  The offset counters of the Extra Path are restarted at the old start value  of the previous skipped frame.</p> <p><b>B)</b> Skip init mode (init_offset_en = 1)  The offset counters of the Extra Path are initialized with the register  contents of the offset counter init registers without any additional forced  software update or automatic config update.</p>
<b>MI_EP_CFG_U PD</b>	4	w	<p><b>Forced Configuration Update</b>  Leads to an immediate update of the shadow registers. Depending on  the two init enable bits in the MI_EP_n_CTRL register (init_offset_en and  init_base_en) the offset counter, base address and buffer size shadow  registers are also updated.</p>
<b>0</b>	1:0, 3, 31:5	r	<p><b>Reserved</b>  Read as 0, should be written with 0.</p>

**Camera and ADC Interface (CIF)****Memory Interface Base Address for Extra Path j Data Buffer Register****MIEP\_j\_BASE\_AD\_INIT (j=0-4)****Memory Interface Base Address for Extra Path j Data Buffer Register( $3608_H + j * 100_H$ )****Application Reset****Value:  $0000\ 0000_H$** 

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EP_BASE_AD_INIT															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EP_BASE_AD_INIT														FIXED_TO_00	
rw															r

Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>EP_BASE_AD_INIT</b>	31:2	rw	<p><b>Extra Path Base Address Init</b>            Base address of Extra Path ring buffer.            Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.</p> <p><i>Note:</i> Set control bit <i>init_base_en</i> before updating so that a forced or automatic update can take effect.</p>

**Memory Interface Size of Extra Path j Data Buffer Register****MIEP\_j\_SIZE\_INIT (j=0-4)****Memory Interface Size of Extra Path j Data Buffer Register( $360C_H + j * 100_H$ )**    **Application Reset Value: 0000 0000\_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								EP_SIZE_INIT							
r								rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EP_SIZE_INIT														FIXED_TO_00	
rw															r

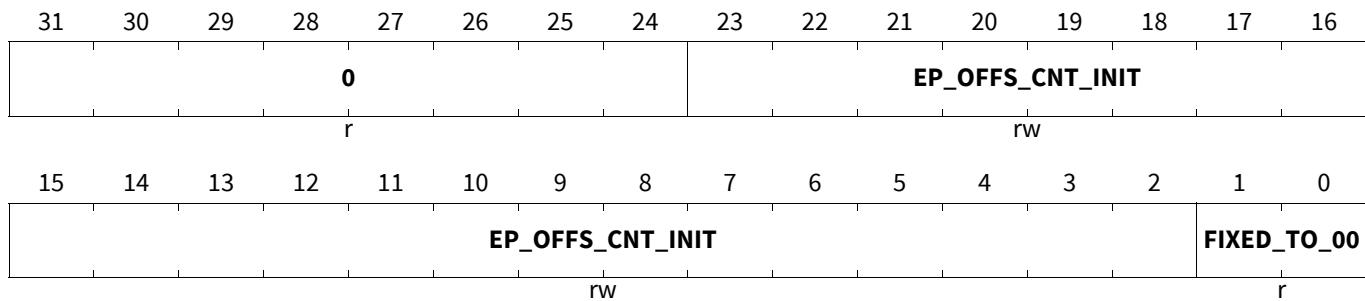
Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>

**Camera and ADC Interface (CIF)**

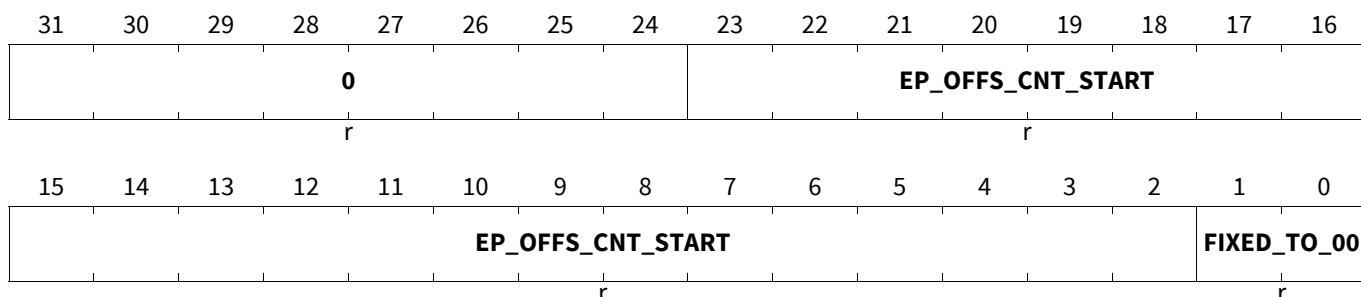
Field	Bits	Type	Description
<b>EP_SIZE_INIT</b>	23:2	rw	<p><b>Extra Path Size Init</b>  Size of Extra Path ring buffer.  Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.</p> <p><i>Note:</i> Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect.</p>
<b>0</b>	31:24	r	<p><b>Reserved</b>  Read as 0, should be written with 0.</p>

**Memory Interface Offset Counter Init Value For Extra Path j Buffer Register****MIEP\_j\_OFFSET\_CNT\_INIT (j=0-4)**

**Memory Interface Offset Counter Init Value For Extra Path j Buffer Register**( $3610_{\text{H}} + j * 100_{\text{H}}$ )      Application  
**Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>EP_OFFSET_CNT_INIT</b>	23:2	rw	<p><b>Extra Path Offset Counter Init</b>  Offset counter init value of Extra Path ring buffer.  Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.</p> <p><i>Note:</i> Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect. Check exceptional handling in skip modes.</p>
<b>0</b>	31:24	r	<p><b>Reserved</b>  Read as 0, should be written with 0.</p>

**Camera and ADC Interface (CIF)****Memory Interface Offset Counter Start Value for Extra Path j Register****MIEP\_j\_OFFSET\_CNT\_START (j=0-4)****Memory Interface Offset Counter Start Value for Extra Path j Register( $3614_H + j * 100_H$ ) Application Reset****Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>EP_OFFSET_CNT_START</b>	23:2	r	<b>Extra Path Offset Counter Start</b> Offset counter value which points to the start address of the previously processed picture. Updated at frame end. Note: A soft reset resets the contents to the reset value.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

**Memory Interface Fill Level Interrupt Offset Value For Extra Path Data Register****MIEP\_j\_IRQ\_OFFSET\_INIT (j=0-4)****Memory Interface Fill Level Interrupt Offset Value For Extra Path Data Register( $3618_H + j * 100_H$ ) Application****Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>EP_IRQ_OFFSET_INIT</b>	23:2	rw	<b>Extra Path Y IRQ Offset Init</b> Reaching this programmed value by the current offset counter for addressing Extra Path buffer leads to generation of fill level interrupt fill_ep_y. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.

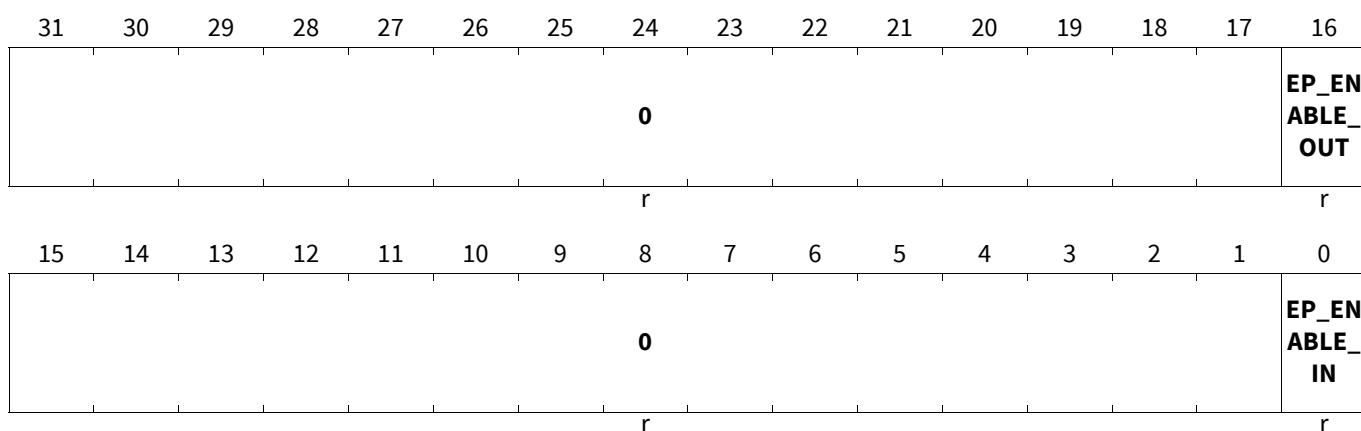
---

**Camera and ADC Interface (CIF)**

Field	Bits	Type	Description
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****26.4.8.4 Extra Path Memory Interface Shadow Registers****Memory Interface Extra Path j Control Internal Shadow Register****MIEP\_j\_CTRL\_SHD (j=0-4)**

**Memory Interface Extra Path j Control Internal Shadow Register(361C<sub>H</sub>+j\*100<sub>H</sub>) Application Reset Value:  
0000 0000<sub>H</sub>**

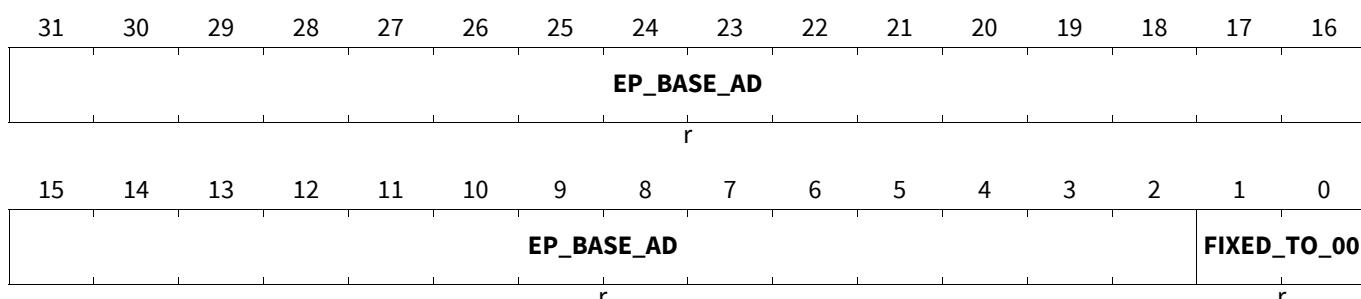


Field	Bits	Type	Description
EP_ENABLE_I N	0	r	<b>Extra Path In Enable</b> Extra Path data used in module MI_IN
EP_ENABLE_ OUT	16	r	<b>Extra Path Out Enable</b> Extra Path used in module MI_OUT
0	15:1, 31:17	r	<b>Reserved</b> Read as 0.

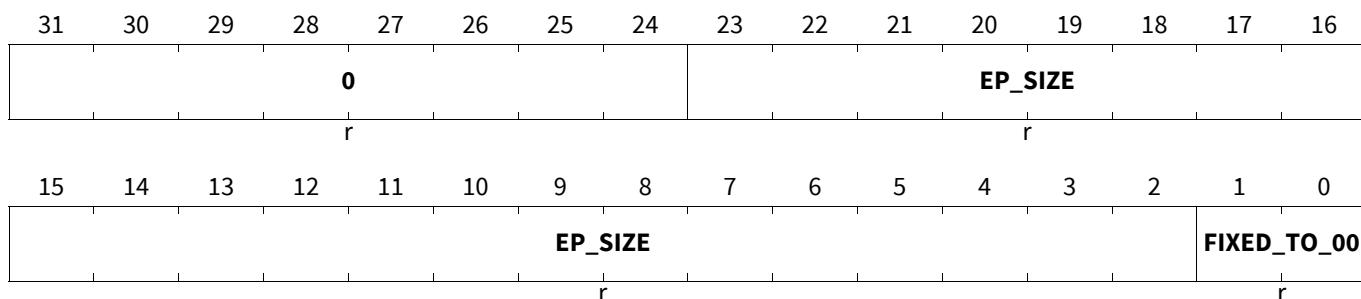
**Memory Interface Base Address Shadow Register for Extra Path j Buffer Register****MIEP\_j\_BASE\_AD\_SHD (j=0-4)**

**Memory Interface Base Address Shadow Register for Extra Path j Buffer Register(3620<sub>H</sub>+j\*100<sub>H</sub>)**

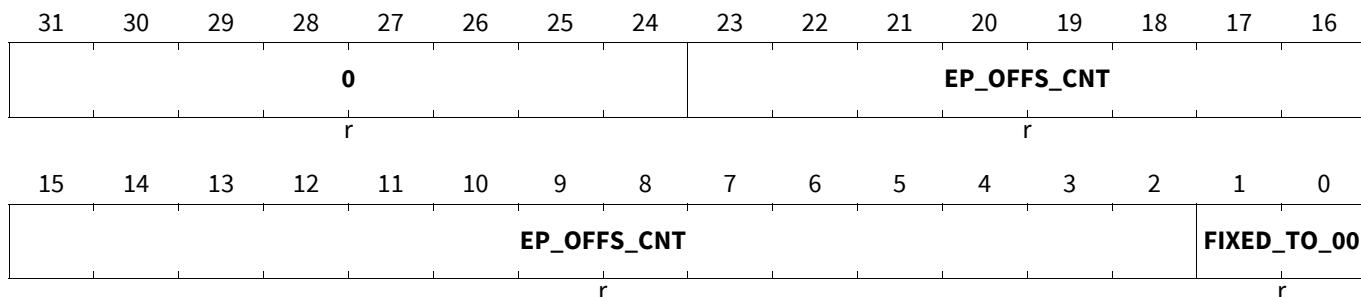
**Application Reset Value: 0000 0000<sub>H</sub>**



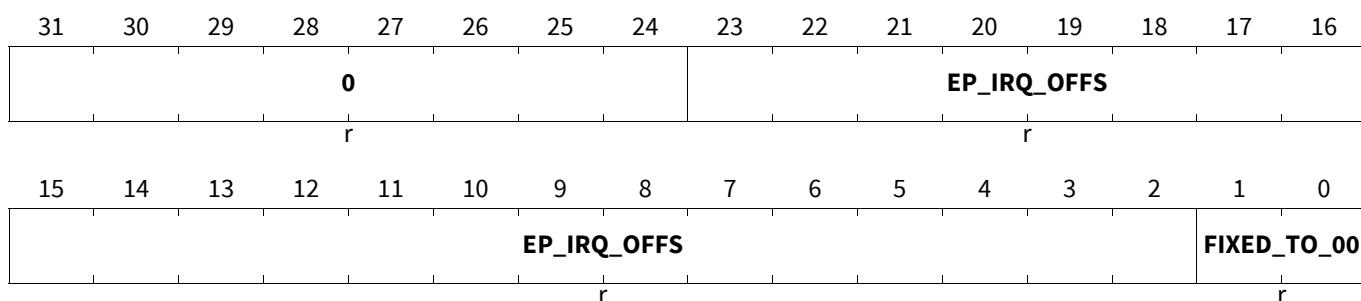
Field	Bits	Type	Description
FIXED_TO_00	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
EP_BASE_AD	31:2	r	<b>Extra Path Base Address</b> Base address of Extra Path ring buffer.

**Camera and ADC Interface (CIF)****Memory Interface Size Shadow Register of Extra Path j Buffer Register****MIEP\_j\_SIZE\_SHD (j=0-4)****Memory Interface Size Shadow Register of Extra Path j Buffer Register( $3624_H+j*100_H$ ) Application Reset****Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>EP_SIZE</b>	23:2	r	<b>Extra Path Size</b> Size of Extra Path ring buffer.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

**Memory Interface Current Offset Counter of Extra Path j Buffer Register****MIEP\_j\_OFFSET\_CNT\_SHD (j=0-4)****Memory Interface Current Offset Counter of Extra Path j Buffer Register( $3628_H+j*100_H$ ) Application Reset****Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to “00” (word aligned value).</b>
<b>EP_OFFSET_CNT</b>	23:2	r	<b>Extra Path Y Offset Counter</b> Current offset counter of Extra Path ringbuffer for address generation.  <i>Note:</i> Soft reset will reset the contents to reset value.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****Memory Interface Shadow Register of Fill Level Interrupt Offset Value For Extra Path j Register****MIEP\_j\_IRQ\_OFFSET\_SHD (j=0-4)****Memory Interface Shadow Register of Fill Level Interrupt Offset Value For Extra Path j Register**(362C<sub>H</sub>+j\*100<sub>H</sub>)      Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
<b>FIXED_TO_00</b>	1:0	r	<b>Bits [1:0] are set to "00" (word aligned value).</b>
<b>EP_IRQ_OFFSET</b>	23:2	r	<b>Extra Path IRQ Offset</b> Reaching this offset value by the current offset counter for addressing Extra Path leads to generation of fill level interrupt fill_ep_y.
<b>0</b>	31:24	r	<b>Reserved</b> Read as 0.

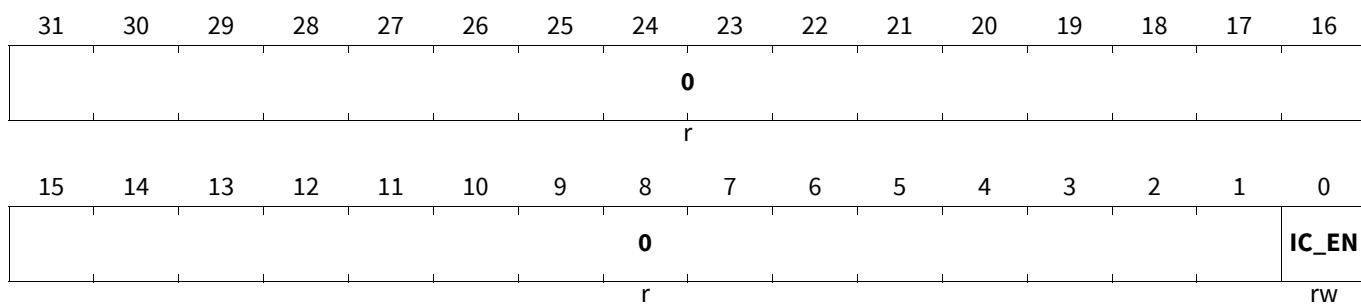
## Camera and ADC Interface (CIF)

## 26.4.8.5 Extra Path Image Cropping Control Registers

The address of each CIF Extra Path Image Cropping register is evaluated as  $\text{CIF\_EP\_IC\_BASE} + \text{Offset}$

## Extra Path i Image Cropping Control Register

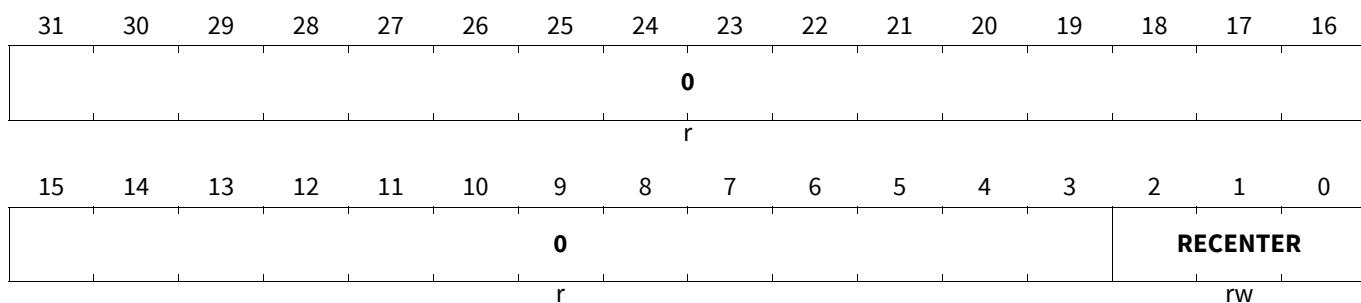
## EP\_i\_IC\_CTRL (i=0-4)

Extra Path i Image Cropping Control Register( $2A00_H + i * 100_H$ )Application Reset Value: 0000 0000<sub>H</sub>

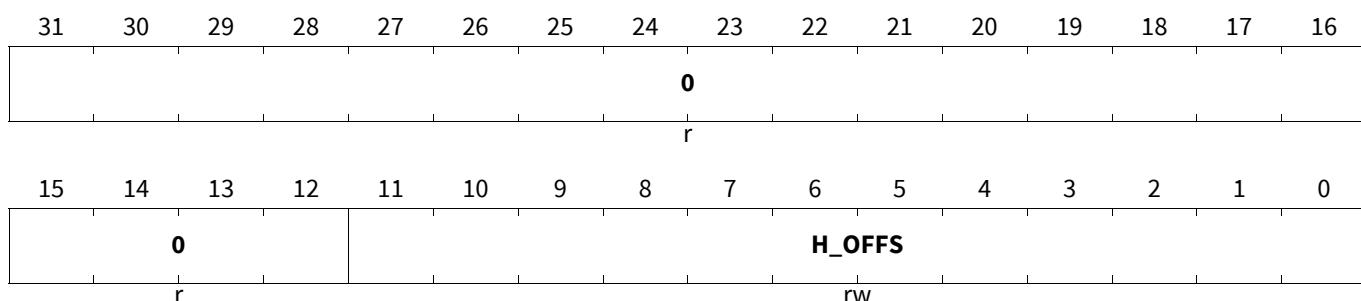
Field	Bits	Type	Description
IC_EN	0	rw	<b>Image Cropping Enable</b> $0_B$ image cropping switched off $1_B$ image cropping switched on
0	31:1	r	<b>Reserved</b> Read as 0.

## Extra Path i Image Cropping Recenter Register

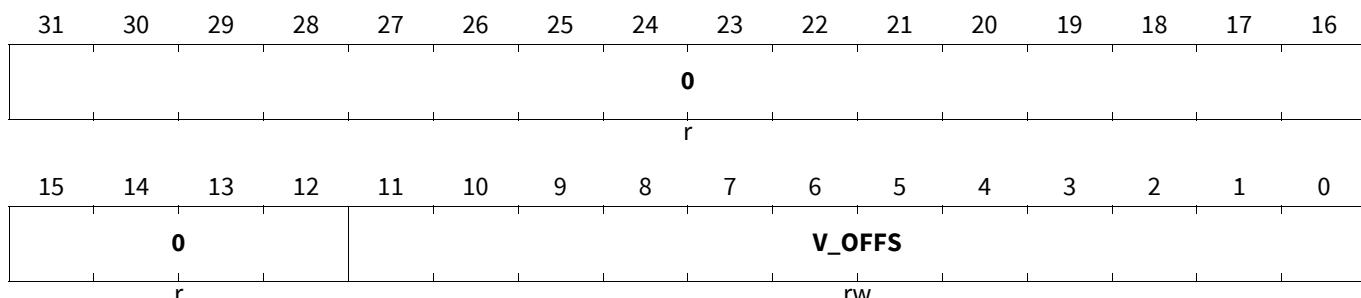
## EP\_i\_IC\_RECENTER (i=0-4)

Extra Path i Image Cropping Recenter Register( $2A04_H + i * 100_H$ )Application Reset Value: 0000 0000<sub>H</sub>

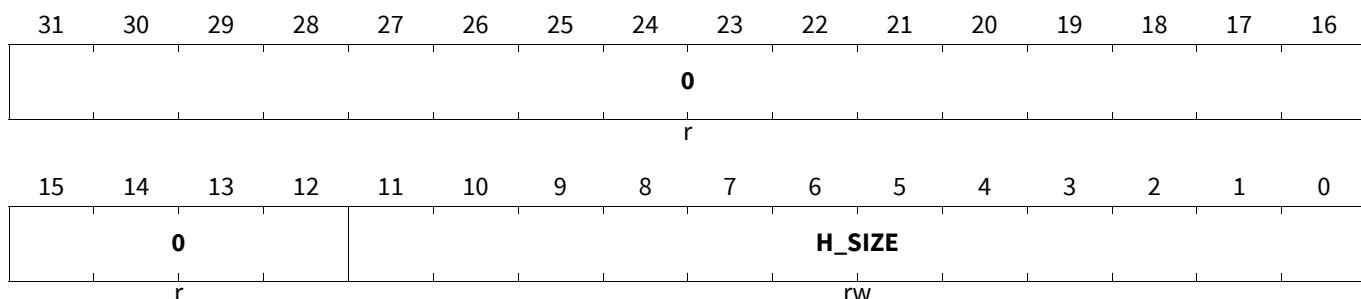
Field	Bits	Type	Description
RECENTER	2:0	rw	<b>Recenter</b> For all other values recentering is active ( $\text{cur\_h/v\_offs-H/V\_OFFS}/2$ ) $000_B$ recenter feature switched off
0	31:3	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****Extra Path i Image Cropping Horizontal Offset of Output Window Register****EP\_i\_IC\_H\_OFFSETS (i=0-4)****Extra Path i Image Cropping Horizontal Offset of Output Window Register( $2A08_H+i*100_H$ )****Application****Reset Value: 0000 0000<sub>H</sub>**

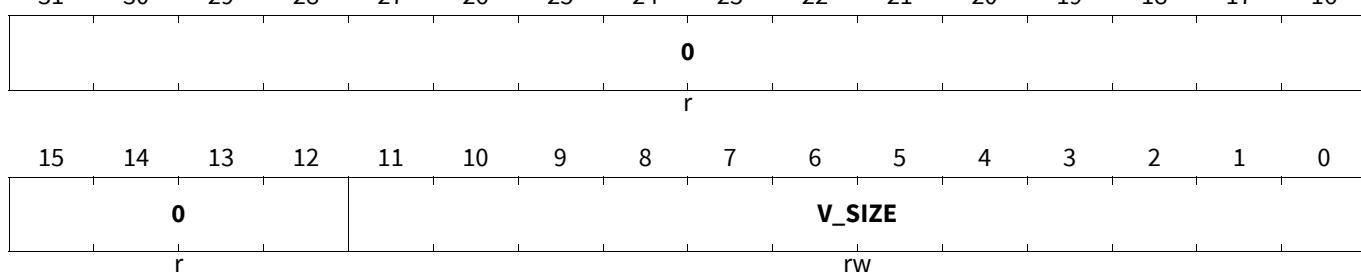
Field	Bits	Type	Description
<b>H_OFFSETS</b>	11:0	rw	<b>Horizontal Picture Offset</b> Horizontal picture offset in pixel
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**Extra Path i Image Cropping Vertical Offset Of Output Window Register****EP\_i\_IC\_V\_OFFSETS (i=0-4)****Extra Path i Image Cropping Vertical Offset Of Output Window Register( $2A0C_H+i*100_H$ )****Application****Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>V_OFFSETS</b>	11:0	rw	<b>Vertical Picture Offset</b> Vertical picture offset in lines
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****Extra Path i Image Cropping Output Horizontal Picture Size Register****EP\_i\_IC\_H\_SIZE (i=0-4)****Extra Path i Image Cropping Output Horizontal Picture Size Register(2A10<sub>H</sub>+i\*100<sub>H</sub>)****Application Reset****Value: 0000 0A28<sub>H</sub>**

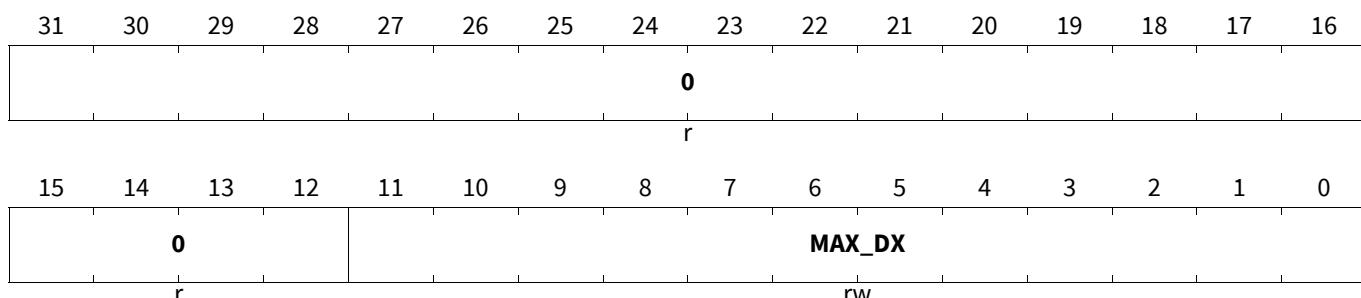
Field	Bits	Type	Description
<b>H_SIZE</b>	11:0	rw	<b>Horizontal Picture Size</b> Horizontal picture size in pixel Only even numbers are accepted, because complete quadruples of YUYV(YCbYCr) are needed for the following modules. If an odd size is programmed the value will be truncated to even size. If ISP_MODE is set to 001 <sub>H</sub> (ITU-R BT.656 YUV) 002 <sub>H</sub> (ITU-R BT.601 YUV) 003 <sub>H</sub> (ITU-R BT.601 Bayer RGB) 005 <sub>H</sub> (ITU-R BT.656 Bayer RGB)
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**Extra Path i Image Cropping Output Vertical Picture Size Register****EP\_i\_IC\_V\_SIZE (i=0-4)****Extra Path i Image Cropping Output Vertical Picture Size Register(2A14<sub>H</sub>+i\*100<sub>H</sub>)****Application Reset Value:****0000 0800<sub>H</sub>**

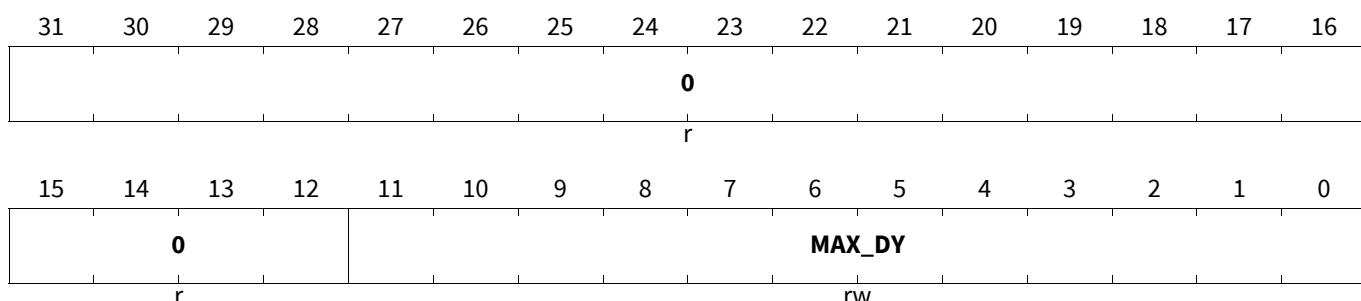
Field	Bits	Type	Description
<b>V_SIZE</b>	11:0	rw	<b>Vertical Picture Size</b> Vertical picture size in lines

**Camera and ADC Interface (CIF)**

Field	Bits	Type	Description
0	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**Extra Path i Image Cropping Maximum Horizontal Displacement Register****EP\_i\_IC\_MAX\_DX (i=0-4)****Extra Path i Image Cropping Maximum Horizontal Displacement Register( $2A18_H+i*100_H$ ) Application Reset****Value: 0000 0000<sub>H</sub>**

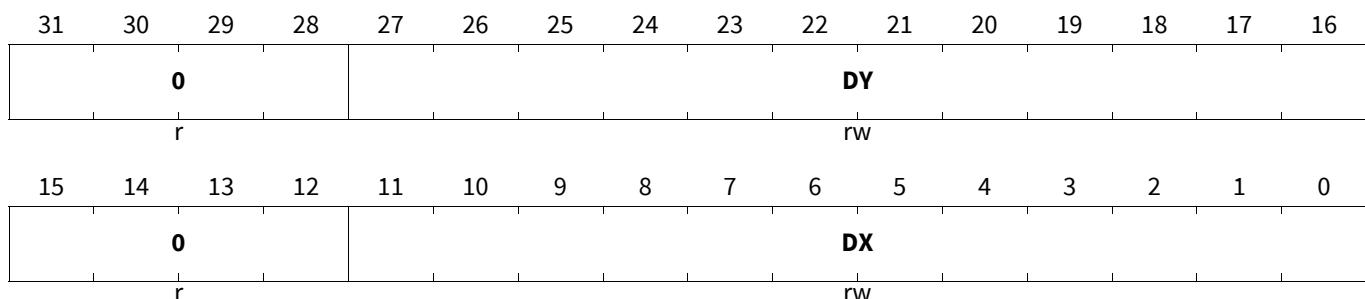
Field	Bits	Type	Description
<b>MAX_DX</b>	11:0	rw	<b>Maximum Horizontal Displacement</b> Maximum allowed accumulated horizontal displacement in pixels
0	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**Extra Path i Image Cropping Maximum Vertical Displacement Register****EP\_i\_IC\_MAX\_DY (i=0-4)****Extra Path i Image Cropping Maximum Vertical Displacement Register( $2A1C_H+i*100_H$ ) Application Reset****Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>MAX_DY</b>	11:0	rw	<b>Maximum Vertical Displacement</b> Maximum allowed accumulated vertical displacement in lines
0	31:12	r	<b>Reserved</b> Read as 0, should be written with 0.

**Camera and ADC Interface (CIF)****Extra Path i Image Cropping Camera Displacement Register****EP\_i\_IC\_DISPLACE (i=0-4)**

**Extra Path i Image Cropping Camera Displacement Register( $2A20_H + i * 100_H$ ) Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>DX</b>	11:0	rw	<b>Camera Displacement</b> Will compensate for horizontal camera displacement of DX pixels in the next frame
<b>DY</b>	27:16	rw	<b>Camera Displacement</b> ISP_IS will compensate for vertical camera displacement of DY pixels in the next frame
<b>0</b>	15:12, 31:28	r	<b>Reserved</b> Read as 0, should be written with 0.

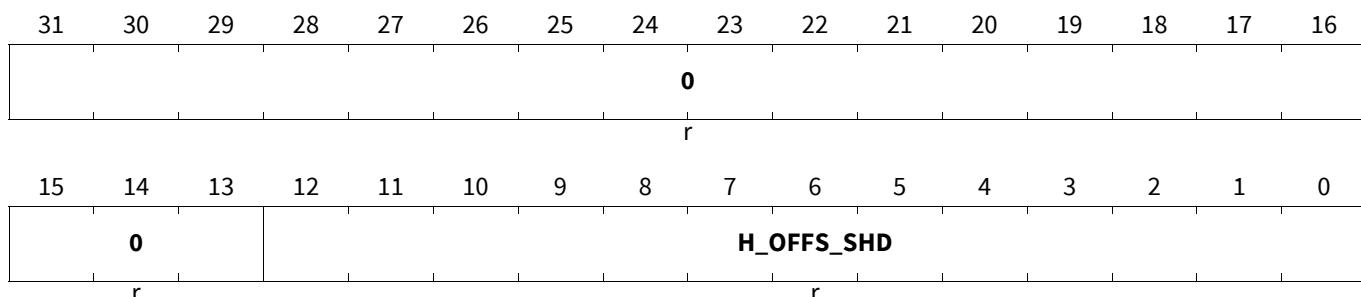
**Camera and ADC Interface (CIF)****26.4.8.6 Extra Path Image Cropping Shadow Registers**

The address of each CIF Extra Path Image Cropping Shadow register is evaluated as CIF\_EP\_IC\_BASE + Offset

**Extra Path i Image Cropping Current Horizontal Offset of Output Window Shadow Register****EP\_i\_IC\_H\_OFFSET\_SHD (i=0-4)**

**Extra Path i Image Cropping Current Horizontal Offset of Output Window Shadow Register(2A24<sub>H</sub>+i\*100<sub>H</sub>)**

**Application Reset Value: 0000 0000<sub>H</sub>**

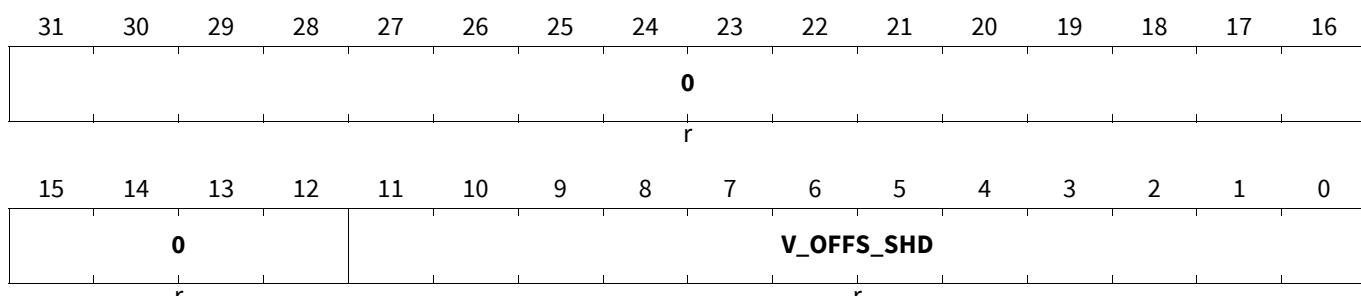


Field	Bits	Type	Description
<b>H_OFFSET_SHD</b>	12:0	r	<b>Horizontal Picture Offset</b> Current horizontal picture offset in lines
<b>0</b>	31:13	r	<b>Reserved</b> Read as 0.

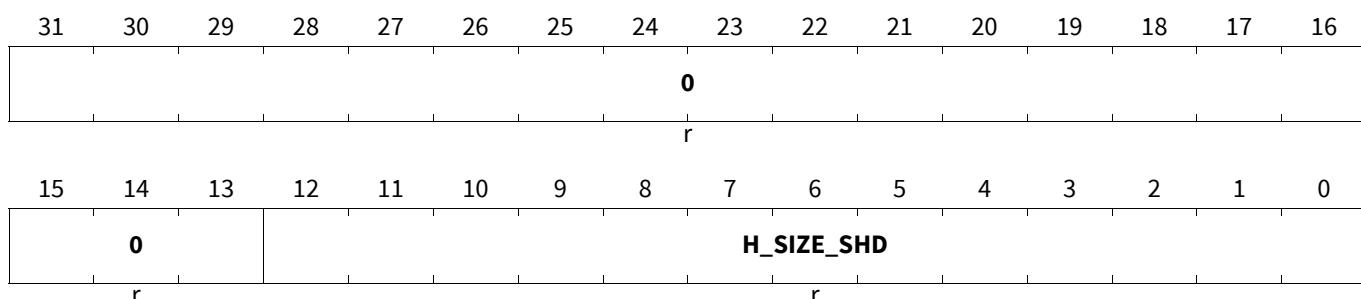
**Extra Path i Image Cropping Current Vertical Offset Of Output Window Shadow Register****EP\_i\_IC\_V\_OFFSET\_SHD (i=0-4)**

**Extra Path i Image Cropping Current Vertical Offset Of Output Window Shadow Register(2A28<sub>H</sub>+i\*100<sub>H</sub>)**

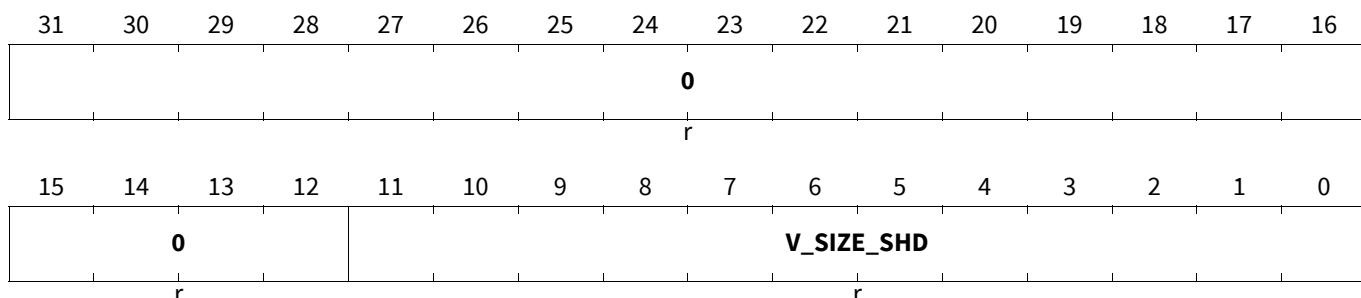
**Application Reset Value: 0000 0000<sub>H</sub>**



Field	Bits	Type	Description
<b>V_OFFSET_SHD</b>	11:0	r	<b>Vertical Picture Offset</b> Current vertical picture offset in lines
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0.

**Camera and ADC Interface (CIF)****Extra Path i Image Cropping Current Output Horizontal Picture Size Shadow Register****EP\_i\_IC\_H\_SIZE\_SHD (i=0-4)****Extra Path i Image Cropping Current Output Horizontal Picture Size Shadow Register(2A2C<sub>H</sub>+i\*100<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>H_SIZE_SHD</b>	12:0	r	<b>Horizontal Picture Size</b> Current horizontal picture size in pixel
<b>0</b>	31:13	r	<b>Reserved</b> Read as 0.

**Extra Path i Image Cropping Current Output Vertical Picture Size Shadow Register****EP\_i\_IC\_V\_SIZE\_SHD (i=0-4)****Extra Path i Image Cropping Current Output Vertical Picture Size Shadow Register(2A30<sub>H</sub>+i\*100<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

Field	Bits	Type	Description
<b>V_SIZE_SHD</b>	11:0	r	<b>Vertical Picture Size</b> Current vertical picture size in lines
<b>0</b>	31:12	r	<b>Reserved</b> Read as 0.

## Camera and ADC Interface (CIF)

## 26.4.9 Debug Path Programming Registers

The address of each CIF Debug Path register is evaluated as CIF\_DEBUG\_PATH\_BASE + Offset.

## 26.4.9.1 Debug Path Control Registers

## Debug Path Control Register

## DP\_CTRL

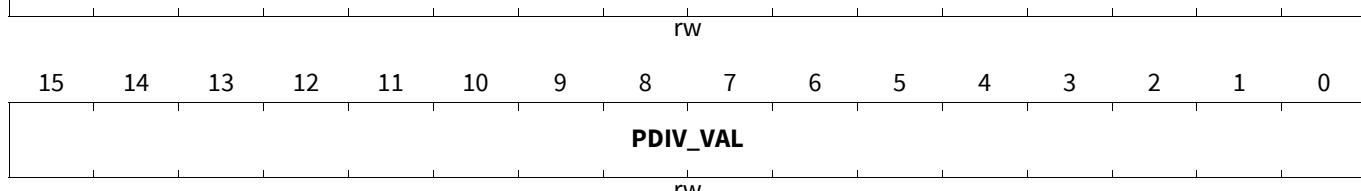
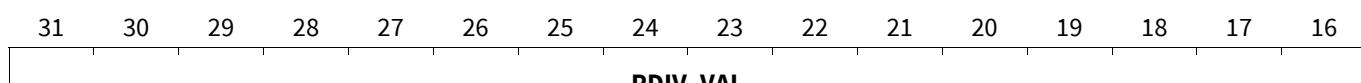
**Debug Path Control Register (2800<sub>H</sub>) Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
							0		UDS8	UDS7	UDS6	UDS5	UDS4	UDS3	UDS2	UDS1
				r					rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
TSC_E N	LNC_E N	FNC_E N	0	RST_P D	RST_T SC	RST_L NC	RST_F NC		0		DP_SEL		DP_EN			
rw	rw	rw	r	w	w	w	w		r		rw		rw		rw	

Field	Bits	Type	Description
DP_EN	0	rw	<b>Debug Path Enable</b> $0_B$ disabled $1_B$ enabled
DP_SEL	3:1	rw	<b>Select Source Path which will be transferred over the Debug Interface</b> $000_B$ Main Path $001_B$ Extra Path 1 $010_B$ Extra Path 2 $011_B$ Extra Path 3 $100_B$ Extra Path 4 $101_B$ Extra Path 5
RST_FNC	8	w	<b>Reset Frame Number Counter</b> $1_B$ Writing to this register bit sets the Counter to zero
RST_LNC	9	w	<b>Reset Line Number Counter</b> $1_B$ Writing to this register bit sets the Counter to zero
RST_TSC	10	w	<b>Reset Timestamp Counter</b> $1_B$ Writing to this register bit sets the Counter to zero
RST_PD	11	w	<b>Reset Predivider Counter</b> $1_B$ Writing to this register bit sets the Counter to zero
FNC_EN	13	rw	<b>Enable/disable Frame Number Counter</b> $0_B$ disable $1_B$ enable
LNC_EN	14	rw	<b>Enable/disable Line Number Counter</b> $0_B$ disable $1_B$ enable

**Camera and ADC Interface (CIF)**

Field	Bits	Type	Description
<b>TSC_EN</b>	15	rw	<b>Enable/disable Timestamp Counter</b> 0 <sub>B</sub> disable 1 <sub>B</sub> enable
<b>UDS1</b>	16	rw	<b>Enable/disable sending of User Defined Symbol 1</b> 0 <sub>B</sub> disable 1 <sub>B</sub> enable
<b>UDS2</b>	17	rw	<b>Enable/disable sending of User Defined Symbol 2</b> 0 <sub>B</sub> disable 1 <sub>B</sub> enable
<b>UDS3</b>	18	rw	<b>Enable/disable sending of User Defined Symbol 3</b> 0 <sub>B</sub> disable 1 <sub>B</sub> enable
<b>UDS4</b>	19	rw	<b>Enable/disable sending of User Defined Symbol 4</b> 0 <sub>B</sub> disable 1 <sub>B</sub> enable
<b>UDS5</b>	20	rw	<b>Enable/disable sending of User Defined Symbol 5</b> 0 <sub>B</sub> disable 1 <sub>B</sub> enable
<b>UDS6</b>	21	rw	<b>Enable/disable sending of User Defined Symbol 6</b> 0 <sub>B</sub> disable 1 <sub>B</sub> enable
<b>UDS7</b>	22	rw	<b>Enable/disable sending of User Defined Symbol 7</b> 0 <sub>B</sub> disable 1 <sub>B</sub> enable
<b>UDS8</b>	23	rw	<b>Enable/disable sending of User Defined Symbol 8</b> 0 <sub>B</sub> disable 1 <sub>B</sub> enable
<b>0</b>	7:4, 12, 31:24	r	<b>Reserved</b> Read as 0, should be written with 0.

**Debug Path Predivider Control Register****DP\_PDIV\_CTRL****Debug Path Predivider Control Register**(2804<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

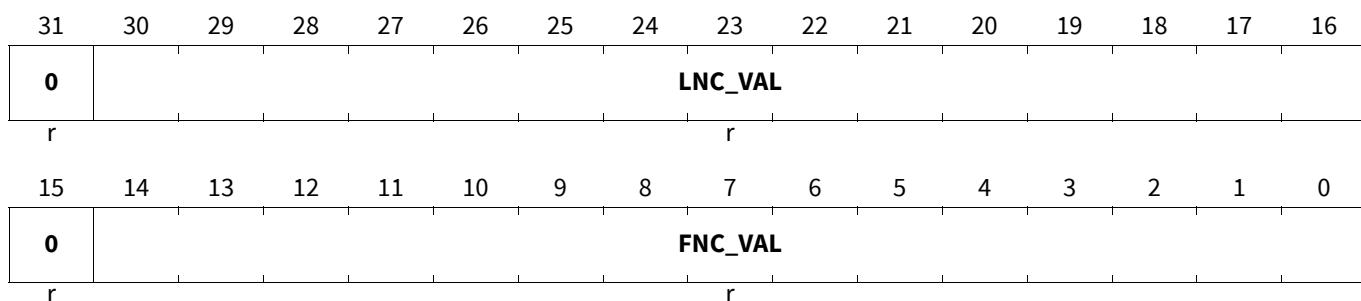
## Camera and ADC Interface (CIF)

Field	Bits	Type	Description
PDIV_VAL	31:0	rw	If the Debug Path and the Timestamp Counter are enabled, the timestamp counter will be increased with every pdiv_val+1 CIF module clock cycle.

## 26.4.9.2 Debug Path Status Registers

## Debug Path Frame/Line Counter Status Register

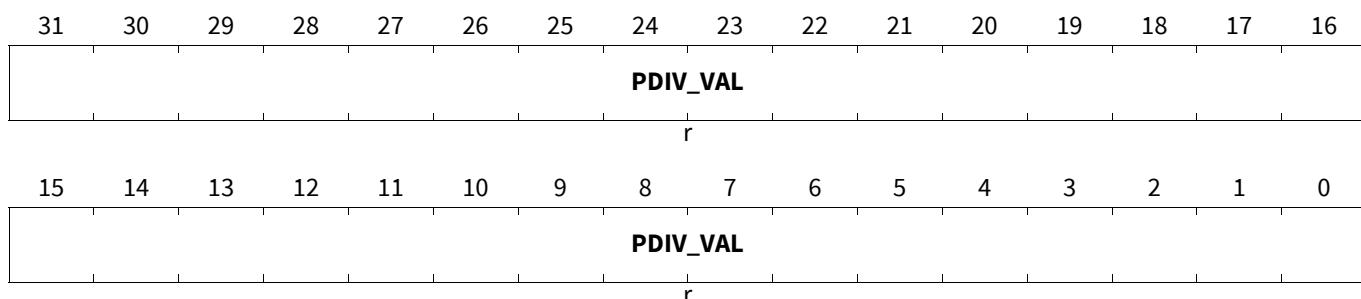
## DP\_FLC\_STAT

Debug Path Frame/Line Counter Status Register(2808<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
FNC_VAL	14:0	r	Returns the current value of the Frame Number Counter
LNC_VAL	30:16	r	Returns the current value of the Line Number Counter
0	15, 31	r	Reserved Read as 0.

## Debug Path Predivider Counter Status Register

## DP\_PDIV\_STAT

Debug Path Predivider Counter Status Register (280C<sub>H</sub>)Application Reset Value: 0000 0000<sub>H</sub>

Field	Bits	Type	Description
PDIV_VAL	31:0	r	Returns the current value of the Predivider Counter.

**Camera and ADC Interface (CIF)****Debug Path Timestamp Counter Status Register****DP\_TSC\_STAT****Debug Path Timestamp Counter Status Register(2810<sub>H</sub>)****Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	TSC_VAL														
r	r														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TSC_VAL														
	r														

Field	Bits	Type	Description
TSC_VAL	29:0	r	Returns the current value of the Timestamp Counter.
0	31:30	r	<b>Reserved</b> Read as 0, should be written with 0.

**26.4.9.3 Debug Path User Defined Symbols Registers****Debug Path User Defined Symbol x Register****DP\_UDS\_x (x=0-7)****Debug Path User Defined Symbol x Register (2814<sub>H</sub>+x\*4)****Application Reset Value: 0000 0000<sub>H</sub>**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	0														
r	r														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	UDS														
r	rw														

Field	Bits	Type	Description
UDS	14:0	rw	User Defined Symbol which may be inserted into debug stream
0	31:15	r	<b>Reserved</b> Read 0, should be written with 0.

**26.5 IO Interfaces****Table 1181 List of CIF Interface Signals**

Interface Signals	I/O	Description
MI_INT	out	<b>CIF MI Service Request</b>
MIEP_INT	out	<b>CIF MI EP Service Request</b>

## Camera and ADC Interface (CIF)

**Table 1181 List of CIF Interface Signals (cont'd)**

Interface Signals	I/O	Description
ISP_INT	out	<b>CIF ISP Service Request</b>
MJPEG_INT	out	<b>CIF MJPEG Service Request</b>
PCLK	in	<b>Sensor Pixel Clock input</b>
D(15:0)	in	<b>sensor pixel data input</b>
Hsync	in	<b>horizontal synchronization signal input</b>
Vsync	in	<b>vertical synchronization signal input</b>

## 26.6 Revision History

**Table 1182 Revision History**

Reference	Change to Previous Version	Comment
<b>V1.4.7</b>	- Translation of the chapter to new documentation format.	
<b>V1.4.8</b>	- Document adjusted to the structure of 6 top-level sub-chapters.  Page 178 <b>Revision History</b> , format reworked. - First-level heading structure adjusted.	
<b>V1.4.9</b>	- Update of document to new template to correct reported problem with page numbering after User Manual build process.  Page 3 <b>Section 26.2.2.1.1, “Camera Interface Example”</b> . New table added for I/O signals  Page 177 <b>Section 26.5, “IO Interfaces”</b> . Updated to align names with chapter text  Page 19 <b>Section 26.3.8.1, “Write to EMEM”</b> . Description of interrupts updated to include name of physical request.  Page 8 <b>Section 26.3.1, “Sub Module ISP”</b> . Details of interrupt connection added  Page 9 <b>Section 26.3.2, “Sub Module Security Watchdog”</b> . Details of interrupt connection added  Page 10 <b>Section 26.3.4, “Sub Module JPEG Encoder”</b> . Details of interrupt connection added  Page 19 <b>Section 26.3.8.1, “Write to EMEM”</b> . Details of interrupt connection added	
<b>V1.4.10</b>	Page 137 <b>Section 26.4.6.2, “Watchdog Interrupt Registers”</b> . Bitfield ordering corrected.  Page 12 <b>Section 26.3.6, “Sub Module Extra Path Units”</b> . Several corrections to grammar and spelling  Page 21 <b>Section 26.3.10, “BBB Slave Interface”</b> . Corrections to spelling and removal of spurious text.	

**Camera and ADC Interface (CIF)****Table 1182 Revision History (cont'd)**

<b>Reference</b>	<b>Change to Previous Version</b>	<b>Comment</b>
Page 21	<b>Section 26.3.12, “Shadow Registers”.</b> Grammar of first sentence improved	
Page 35	<b>Section 26.3.14.7.1, “ISP Programming”</b> , subsection “Data Mode”. Incorrectly inserted paragraph break removed after “if the transfer indicator line”.	

**V1.4.11**

Page 21	<b>Section 26.3.10, “BBB Slave Interface”.</b> Spelling mistake. “ENDINT” corrected to “ENDINIT”.	
All	Cosmetic Update of Tables. No customer relevant updates.	

**V1.4.12**

Page 52	<b>Section 26.4.</b> Register summary table updated to add 32 bit access restriction to each register	
---------	---	--

## Revision history

Document version	Date of release	Description of changes
V1.5.0	2020-04	<ul style="list-style-type: none"> <li>Version comparison table updated.</li> <li>For further changes see respective revision history of each chapter. The version comparison table below gives an overview.</li> </ul>
V1.4.0	2019-12	<ul style="list-style-type: none"> <li>Added TC3Ax appendix as target specification.</li> <li>Version comparison table updated.</li> <li>For further changes see respective revision history of each chapter. The version comparison table below gives an overview.</li> </ul>
V1.3.0	2019-09	<ul style="list-style-type: none"> <li>Added additional device TC3Ax to AURIX™ TC3xx set of documentation.</li> <li>Version comparison table updated.</li> <li>For further changes see respective revision history of each chapter. The version comparison table below gives an overview.</li> </ul>
V1.2.0	2019-04	<ul style="list-style-type: none"> <li>Added additional device TC3Ex to AURIX™ TC3xx set of documentation.</li> <li>Version comparison table updated.</li> <li>For further changes see respective revision history of each chapter. The version comparison table below gives an overview.</li> </ul>
V1.1.0	2019-01	<ul style="list-style-type: none"> <li>Power Management System for Low-End (PMSLE) added.</li> <li>TC33x and TC33xEXT added.</li> <li>Changes in connectivity tables.</li> <li>Detailed Revision History contained in each chapter.</li> </ul>
V1.0.0	2018-08	<ul style="list-style-type: none"> <li>First revision of the User's Manual.</li> <li>Detailed OCDS and MiniMCDS information is not contained. Available under NDA. Overview available in Introduction chapter.</li> <li>Detailed Revision History contained in each chapter.</li> </ul>

### Version comparison table for AURIX™ TC3xx Family Specification

Chapter name	UM V1.4.0 chapter version	UM V1.5.0 chapter version	Content changes
Introduction	V1.6.1	V1.6.1	No
• OCDS	V3.1.14	V3.1.14	No
• ED	V1.0.5	V1.0.5	No
• SOTA	V1.0.4	V1.0.4	No
MEMMAP	V0.1.18	V0.1.19	Yes, see chapter revision history
FW	V1.1.0.1.17	V1.1.0.1.17	No
SRI Fabric	V1.1.16	V1.1.16	No
• FPI, BCU	V1.2.8	V1.2.8	No
CPU	V1.1.19	V1.1.20	<b>Yes, see chapter revision history</b>
NVM Subsystem	V2.0.4	V2.0.5	<b>Yes, see chapter revision history</b>

Chapter name	UM V1.4.0 chapter version	UM V1.5.0 chapter version	Content changes
• DMU	V2.0.10	V2.0.11	<b>Yes, see chapter revision history</b>
• PFI	V2.0.1	V2.0.1	No
• NVM	V2.0.5	V2.0.6	<b>Yes, see chapter revision history</b>
• UCB	V2.0.21	V2.0.22	<b>Yes, see chapter revision history</b>
LMU	V3.1.16	V3.1.16	No
DAM	V1.3.11	V1.3.11	No
SCU	V2.1.24	V2.1.25	<b>Yes, see chapter revision history</b>
CCU	V2.0.30	V2.0.30	No
PMS	V2.2.31	V2.2.32	<b>Yes, see chapter revision history</b>
PMSLE	V1.0.4	V1.0.5	<b>Yes, see chapter revision history</b>
MTU	V7.4.10	V7.4.11	<b>Yes, see chapter revision history</b>
PORTS	V1.8.21	V1.8.21	No
SMU	V4.0.20	V4.0.21	No functional changes
INT	V1.2.10	V1.2.10	No
FCE	V4.2.9	V4.2.9	No
DMA	V0.1.17	V0.1.17	No
SPU	V1.1.24	V1.1.24	No
SPU2	V2.0.1	V2.0.2	<b>Yes, see chapter revision history</b>
BITMGR	V1.0.4	V1.0.6	<b>Yes, see chapter revision history</b>
SPULCKSTP	V1.2.5	V1.2.5	No
EMEM	V1.4.2	V1.4.3	<b>Yes, see chapter revision history</b>
RIF	V1.0.39	V1.0.40	<b>Yes, see chapter revision history</b>
HSPDM	V0.7.9	V0.7.9	No
CIF	V1.4.12	V1.4.12	No
STM	V9.2.4	V9.2.4	No
GTM	V2.2.20	V2.2.21	<b>Yes, see chapter revision history</b>
CCU6	V3.0.0	V3.0.0	No
GPT12	V3.0.2	V3.0.2	No
CONVCTRL	V3.0.1	V3.0.1	No
EVADC	V3.0.2	V3.0.3	<b>Yes, see chapter revision history</b>
EDSADC	V3.0.3	V3.0.4	<b>Yes, see chapter revision history</b>
I2C	V2.3.5	V2.3.6	<b>Yes, see chapter revision history</b>
HSSL	V3.0.17	V3.0.18	No functional changes
• HSCT	V2.3.13	V2.3.14	<b>Yes, see chapter revision history</b>
ASCLIN	V3.2.8	V3.2.8	No
QSPI	V3.0.20	V3.0.20	No
MSC	V5.0.11	V5.0.11	No
SENT	V2.1.10	V2.1.10	No

Chapter name	UM V1.4.0 chapter version	UM V1.5.0 chapter version	Content changes
MCMCAN	V1.19.11	V1.19.12	<b>Yes, see chapter revision history</b>
E-Ray	V3.2.10	V3.2.10	No
PSI5	V1.17.12	V1.17.12	No
PSI5-S	V1.12.10	V1.12.10	No
GETH	V1.3.13	V1.3.13	No
EBU	V4.0.11	V4.0.12	<b>Yes, see chapter revision history</b>
SDMMC	V1.0.17	V1.0.17	No
HSM	V2.3.9	V2.3.9	No
IOM	V2.1.15	V2.1.15	No
SCR and all subchapters	V4.0.4	V4.0.4	No

### **Known Issues, not considered in Chapter Revision Histories**

**Note:** Common entries in chapter revision histories like “Connection Tables changed” or “Register and Connectivity Tables updated” mean formal changes only. No functional changes.

<b>Chapter name</b>	<b>Chapter version</b>	<b>Family Specification content changes</b>

### **Trademarks of Infineon Technologies AG**

µHVIC™, µPIM™, µPFC™, AU-ConvertIR™, AURIX™, C166™, CanPAK™, CIPOS™, CIPURSE™, CoolDP™, CoolGaN™, COOLiR™, CoolMOS™, CoolSET™, CoolSiC™, DAVE™, DI-POL™, DirectFET™, DrBlade™, EasyPIM™, EconoBRIDGE™, EconoDUAL™, EconoPACK™, EconoPIM™, EiceDRIVER™, euepc™, FCOST™, GaNpowIR™, HEXFET™, HITFET™, HybridPACK™, iMOTION™, IRAM™, ISOFACE™, IsoPACK™, LEDrivelR™, LITIX™, MIPAQ™, ModSTACK™, my-d™, NovalithIC™, OPTIGA™, OptiMOS™, ORIGA™, PowIRaudio™, PowIRStage™, PrimePACK™, PrimeSTACK™, PROFET™, PRO-SIL™, RASIC™, REAL3™, SmartLEWIS™, SOLID FLASH™, SPOCT™, StronglRFET™, SuplRBuck™, TEMPFET™, TRENCHSTOP™, TriCore™, UHVIC™, XHP™, XMCT™.

Trademarks updated November 2015

### **Other Trademarks**

All referenced product or service names and trademarks are the property of their respective owners.

#### **Edition 2020-04**

#### **Published by**

**Infineon Technologies AG  
81726 Munich, Germany**

**© 2020 Infineon Technologies AG.  
All Rights Reserved.**

**Do you have a question about any aspect of this document?**

Email: [erratum@infineon.com](mailto:erratum@infineon.com)

#### **Document reference**

### **IMPORTANT NOTICE**

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenheitsgarantie").

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on technology, delivery terms and conditions and prices, please contact the nearest Infineon Technologies Office ([www.infineon.com](http://www.infineon.com)).

### **WARNINGS**

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.