nRF9160

Product Specification

v1.1

4418_1315 v1.1 / 2019-10-16

nRF9160 features

Features:

Microcontroller:

- ARM® Cortex® -M33
  - 243 EEMBC CoreMark score running from flash memory
  - Data watchpoint and trace (DWT), embedded trace macrocell (ETM), and instrumentation trace macrocell (ITM)
  - Serial wire debug (SWD)
  - Trace port
- 1 MB flash
- 256 kB low leakage RAM
- ARM® Trustzone®
- ARM® Cryptocell 310
- Up to 4x SPI master/slave with EasyDMA
- Up to 4x I2C compatible two-wire master/slave with EasyDMA
- Up to 4x UART (CTS/RTS) with EasyDMA
- I2S with EasyDMA
- Digital microphone interface (PDM) with EasyDMA
- 4x pulse width modulator (PWM) unit with EasyDMA
- 12-bit, 200 ksps ADC with EasyDMA - eigth configurable channels with programmable gain
- 3x 32-bit timer with counter mode
- 2x real-time counter (RTC)
- Programmable peripheral interconnect (PPI)
- 32 general purpose I/O pins
- Single supply voltage: 3.0 – 5.5 V
- All necessary clock sources integrated
- Package: 10 × 16 x 1.04 mm LGA

LTE modem:

- Transceiver and baseband
- 3GPP LTE release 13 Cat-M1 and Cat-NB1 compliant
  - 3GPP release 13 coverage enhancement
- 3GPP LTE release 14 Cat-NB2 compliant
- GPS receiver
  - GPS L1 C/A supported
- RF transceiver for global coverage
  - Up to 23 dBm output power
  - -108 dBm sensitivity (LTE-M) for low band, -107 dBm for mid band
  - Single 50 Ω antenna interface
- LTE band support in hardware:
  - Cat-M1: B1, B2, B3, B4, B5, B8, B12, B13, B14, B17, B18, B19, B20, B25, B26, B28, B66
  - Cat-NB1/NB2: B1, B2, B3, B4, B5, B8, B12, B13, B17, B18, B20, B25, B26, B28, B66
- Supports SIM and eSIM with an ETSI TS 102 221 compatible UICC interface
- Power saving features: DRX, eDRX, PSM
- IP v4/v6 stack
- Secure socket (TLS/DTLS) API

Current consumption @ 3.7 V:

- Power saving mode (PSM) floor current: 4 µA
  - Application core idle: 1.8 µA
  - Modem: 2.2 µA
- eDRX @ 82.91s: 19 µA

Applications:

- Sensor networks
- Logistics and asset tracking
- Smart energy
- Smart building automation
- Smart agriculture
- Industrial
- Retail and monitor devices
- Medical devices
- Wearables

Contents

1 Revision history

Date	Version	Description
October 2019	1.1	Updated: • Debug and trace Overview on page 371: Added debug access port. Updated SDK version • Memory on page 20: Added a reference • Pin assignments on page 386: Several updates • Operating conditions on page 391: Changed chapter name. Updated MAGPIO values. Updated VDD_GPIO restrictions. • Ordering information on page 393: Updated Product options Added: • Reference circuitry on page 389
May 2019	1.0	First release

2 About this document

This document is organized into chapters that are based on the modules and peripherals available in the IC.

2.1 Document status

The document status reflects the level of maturity of the document.

Document name	Description
Objective Product Specification (OPS)	Applies to document versions up to 1.0. This document contains target specifications for product development.
Product Specification (PS)	Applies to document versions 1.0 and higher. This document contains final product specifications. Nordic Semiconductor ASA reserves the right to make changes at any time without notice in order to improve design and supply the best possible product.

Table 1: Defined document names

2.2 Peripheral chapters

Every peripheral has a unique capitalized name or an abbreviation of its name, e.g. TIMER, used for identification and reference. This name is used in chapter headings and references, and it will appear in the ARM® Cortex® Microcontroller Software Interface Standard (CMSIS) hardware abstraction layer to identify the peripheral.

The peripheral instance name, which is different from the peripheral name, is constructed using the peripheral name followed by a numbered postfix, starting with 0, for example, TIMER0. A postfix is normally only used if a peripheral can be instantiated more than once. The peripheral instance name is also used in the CMSIS to identify the peripheral instance.

The chapters describing peripherals may include the following information:

- A detailed functional description of the peripheral
- Register configuration for the peripheral
- Electrical specification tables, containing performance data which apply for the operating conditions described in Peripheral chapters on page 10.

2.3 Register tables

Individual registers are described using register tables. These tables are built up of two sections. The first three colored rows describe the position and size of the different fields in the register. The following rows describe the fields in more detail.

2.3.1 Fields and values

The **Id (Field Id)** row specifies the bits that belong to the different fields in the register. If a field has enumerated values, then every value will be identified with a unique value id in the **Value Id** column.

A blank space means that the field is reserved and read as undefined, and it also must be written as `0` to secure forward compatibility. If a register is divided into more than one field, a unique field name is specified for each field in the **Field** column. The **Value Id** may be omitted in the single-bit bit fields when values can be substituted with a Boolean type enumerator range, e.g. true/false, disable(d)/enable(d), on/off, and so on.

Values are usually provided as decimal or hexadecimal. Hexadecimal values have a `0x` prefix, decimal values have no prefix.

The **Value** column can be populated in the following ways:

- Individual enumerated values, for example 1, 3, 9.
- Range of values, e.g. [0..4], indicating all values from and including 0 and 4.
- Implicit values. If no values are indicated in the **Value** column, all bit combinations are supported, or alternatively the field's translation and limitations are described in the text instead.

If two or more fields are closely related, the **Value Id**, **Value**, and **Description** may be omitted for all but the first field. Subsequent fields will indicate inheritance with '..'.

A feature marked **Deprecated** should not be used for new designs.

2.3.2 Permissions

Different fields in a register might have different access permissions enforced by hardware.

The access permission for each register field is documented in the **Access** column in the following ways:

Access	Description	Hardware behavior
RO	Read-only	Field can only be read. A write will be ignored.
WO	Write-only	Field can only be written. A read will return an undefined value.
RW	Read-write	Field can be read and written multiple times.
W1	Write-once	Field can only be written once per reset. Any subsequent write will be ignored. A read will return an undefined value.
RW1	Read-write-once	Field can be read multiple times, but only written once per reset. Any subsequent write will be ignored.

Table 2: Register field permission schemes

2.4 Registers

Register	Offset	Security	Description
DUMMY	0x514		Example of a register controlling a dummy feature

Table 3: Register overview

2.4.1 DUMMY

Address offset: 0x514

Example of a register controlling a dummy feature

Bit number	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
ID					D	D	D	D						C	C	C								B							A	A
Reset 0x00050002	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**

ID	Acce	Field	Value ID	Value	Description	
A	RW	FIELD_A			Example of a read-write field with several enumerated values	
			Disabled	0	The example feature is disabled	
			NormalMode	1	The example feature is enabled in normal mode	
			ExtendedMode	2	The example feature is enabled along with extra functionality	
B	RW	FIELD_B			Example of a deprecated read-write field	Deprecated
			Disabled	0	The override feature is disabled	
			Enabled	1	The override feature is enabled	
C	RW	FIELD_C			Example of a read-write field with a valid range of values	
			ValidRange	[2..7]	Example of allowed values for this field	
D	RW	FIELD_D			Example of a read-write field with no restriction on the values	

3 Product overview

3.1 Introduction

The nRF9160 is a low-power cellular IoT (Internet of Things) solution, integrating an ARM® Cortex®-M33 processor with advanced security features, a range of peripherals, as well as a complete LTE modem compliant with 3GPP LTE release 13 Cat-M1 and Cat-NB1, and 3GPP LTE release 14 Cat-NB1 and Cat-NB2 standards.

The ARM® Cortex-M33 processor is exclusively for user application software, and it offers 1 MB of flash and 256 kB of RAM dedicated to this use. The M33 application processor shares the power, clock and peripheral architecture with Nordic Semiconductor nRF51 and nRF52 Series of PAN/LAN SoCs, ensuring minimal porting efforts.

The peripheral set offers a variety of analog and digital functionality enabling single-chip implementation of a wide range of cellular IoT (Internet of Things) applications. ARM® TrustZone® technology, Cryptocell 310 and supporting blocks for system protection and key management, are embedded to enable advanced security needed for IoT applications.

The LTE modem integrates a very flexible transceiver that in hardware supports frequency range from 700 to 2200 MHz (through a single 50 Ω antenna pin), and a baseband processor handling LTE Cat-M1/NB1/NB2 protocol layers L1-L3 as well as IP upper layers offering secure socket API for the application. The modem is supported by pre-qualified software builds available for free from Nordic Semiconductor.

On specific nRF9160 device variants, the LTE modem supports A-GPS operation during sleep intervals in the LTE operation (RRC idle and PSM modes).

3.2 Block diagram

This block diagram illustrates the overall system. Arrows with white heads indicate signals that share physical pins with other signals.

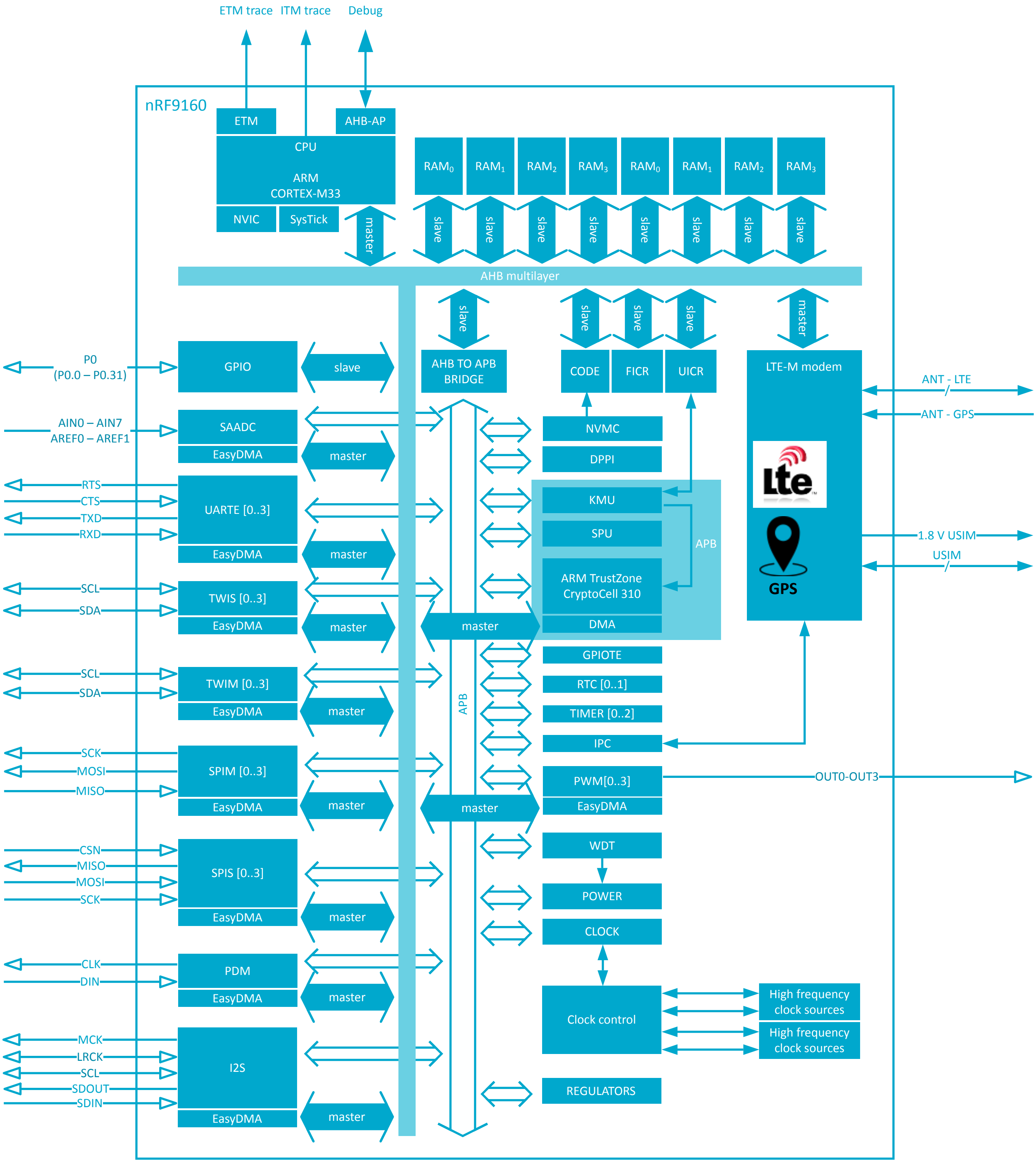

Figure 1: Block diagram

3.3 Peripheral interface

Peripherals are controlled by the CPU through configuration registers, as well as task and event registers. Task registers are inputs, enabling the CPU and other peripherals to initiate a functionality. Event registers are outputs, enabling a peripheral to trigger tasks in other peripherals and/or the CPU by tying events to CPU interrupts.

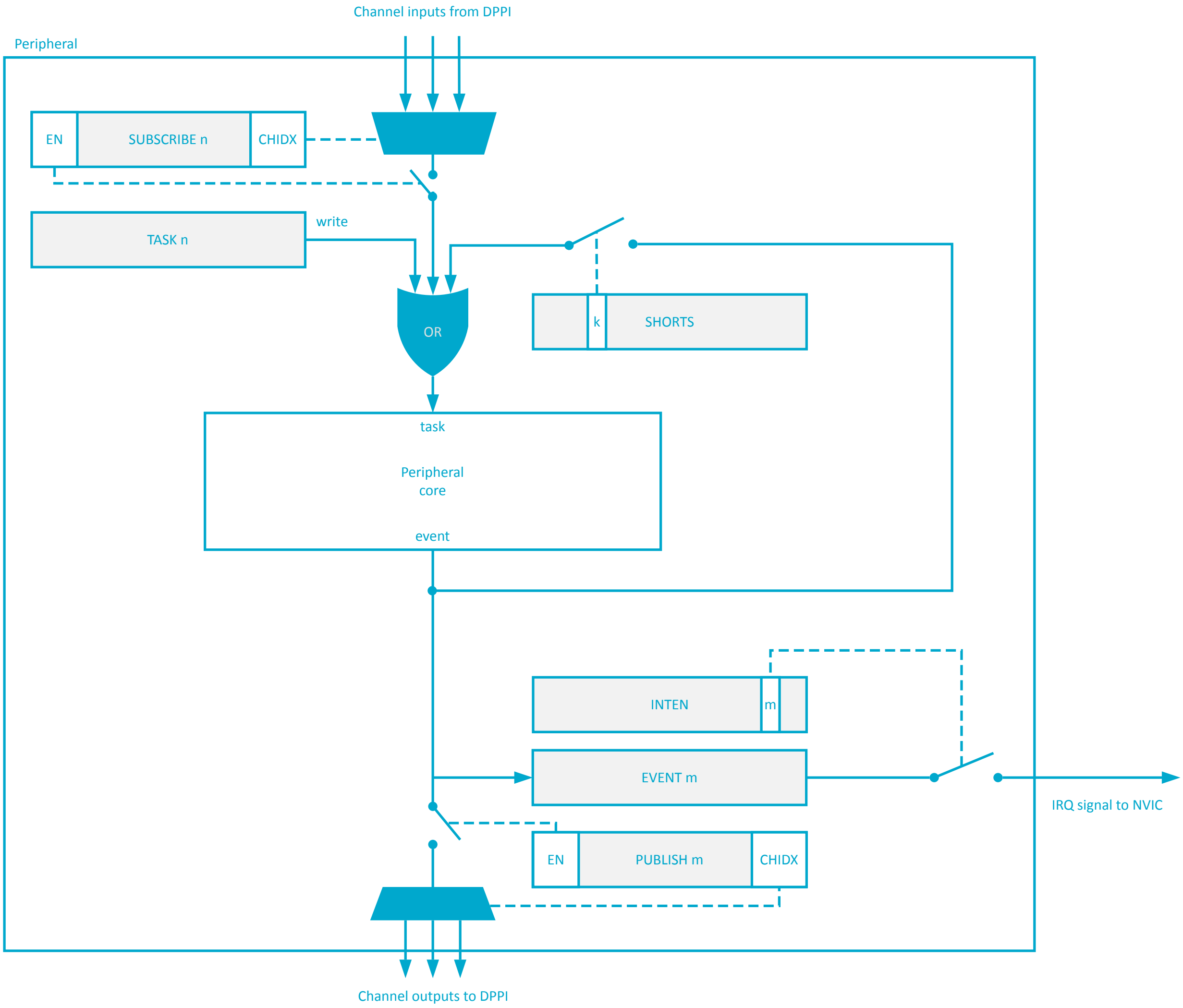

Figure 2: Tasks, events, shortcuts, publish, subscribe and interrupts

The distributed programmable peripheral interconnect (DPPI) feature enables peripherals to connect events to tasks without CPU intervention.

Note: For more information on DPPI and the DPPI channels, see DPPI - Distributed programmable peripheral interconnect on page 82.

3.3.1 Peripheral ID

Every peripheral is assigned a fixed block of 0x1000 bytes of address space, which is equal to 1024 x 32 bit registers.

See Instantiation on page 23 for more information about which peripherals are available and where they are located in the address map.

There is a direct relationship between peripheral ID and base address. For example, a peripheral with base address 0x40000000 is assigned ID=0, a peripheral with base address 0x40001000 is assigned ID=1, and a peripheral with base address 0x4001F000 is assigned ID=31.

Peripherals may share the same ID, which may impose one or more of the following limitations:

- Some peripherals share some registers or other common resources.
- Operation is mutually exclusive. Only one of the peripherals can be used at a time.

- Switching from one peripheral to another must follow a specific pattern (disable the first, then enable the second peripheral).

3.3.2 Peripherals with shared ID

In general (with the exception of ID 0), peripherals sharing an ID and base address may not be used simultaneously. The user can only enable one peripheral at the time on this specific ID.

When switching between two peripherals sharing an ID, the user should do the following to prevent unwanted behavior:

- Disable the previously used peripheral.
- Disable any publish/subscribe connection to the DPPI system for the peripheral that is being disabled.
- Clear all bits in the INTEN register, i.e. INTENCLR = 0xFFFFFFFF.
- Explicitly configure the peripheral that you are about to enable, and do not rely on configuration values that may be inherited from the peripheral that was disabled.
- Enable the now configured peripheral.

See which peripherals are sharing ID in Instantiation on page 23.

3.3.3 Peripheral registers

Most peripherals feature an ENABLE register. Unless otherwise is specified in the chapter, the peripheral registers must be configured before enabling the peripheral.

PSEL registers need to be set before a peripheral is enabled or started. Updating PSEL registers while the peripheral is running has no effect. In order to connect a peripheral to a different GPIO, the peripheral must be disabled, the PSEL register updated and the peripheral re-enabled. It takes four CPU cycles between the PSEL register update and the connection between a peripheral and a GPIO becoming effective.

Note that the peripheral must be enabled before tasks and events can be used.

Most of the register values are lost during System OFF or when a reset is triggered. Some registers will retain their values in System OFF or for some specific reset sources. These registers are marked as retained in the register description for a given peripheral. For more info on these retained registers' behavior, see chapter Reset on page 54.

3.3.4 Bit set and clear

Registers with multiple single-bit bit fields may implement the set-and-clear pattern. This pattern enables firmware to set and clear individual bits in a register without having to perform a read-modify-write operation on the main register.

This pattern is implemented using three consecutive addresses in the register map, where the main register is followed by dedicated SET and CLR registers (in that exact order).

The SET register is used to set individual bits in the main register, while the CLR register is used to clear individual bits in the main register. Writing 1 to a bit in SET or CLR register will set or clear the same bit in the main register respectively. Writing 0 to a bit in SET or CLR register has no effect. Reading the SET or CLR register returns the value of the main register.

> **Note:** The main register may not be visible and hence not directly accessible in all cases.

3.3.5 Tasks

Tasks are used to trigger actions in a peripheral, for example to start a particular behavior. A peripheral can implement multiple tasks with each task having a separate register in that peripheral's task register group.

A task is triggered when firmware writes 1 to the task register, or when the peripheral itself or another peripheral toggles the corresponding task signal. See the figure Tasks, events, shortcuts, publish, subscribe and interrupts on page 15.

3.3.6 Events

Events are used to notify peripherals and the CPU about events that have happened, for example a state change in a peripheral. A peripheral may generate multiple events, where each event has a separate register in that peripheral's event register group.

An event is generated when the peripheral itself toggles the corresponding event signal, and the event register is updated to reflect that the event has been generated (see figure Tasks, events, shortcuts, publish, subscribe and interrupts on page 15). An event register is only cleared when firmware writes 0 to it. Events can be generated by the peripheral even when the event register is set to 1.

3.3.7 Publish / Subscribe

Events and tasks from different peripherals can be connected together through the DPPI. See Tasks, events, shortcuts, publish, subscribe and interrupts on page 15. This is done through publish / subscribe registers in each peripheral. An event can be published onto a DPPI channel by configuring the event's PUBLISH register. Similarly a task can subscribe to a DPPI channel by configuring the task's SUBSCRIBE register.

See DPPI - Distributed programmable peripheral interconnect on page 82 for details.

3.3.8 Shortcuts

A shortcut is a direct connection between an event and a task within the same peripheral. If a shortcut is enabled, the associated task is automatically triggered when its associated event is generated.

Using shortcuts is equivalent to making the connection outside the peripheral and through the DPPI. However, the propagation delay when using shortcuts is usually shorter than the propagation delay through the DPPI.

Shortcuts are predefined, which means that their connections cannot be configured by firmware. Each shortcut can be individually enabled or disabled through the shortcut register, one bit per shortcut, giving a maximum of 32 shortcuts for each peripheral.

3.3.9 Interrupts

All peripherals support interrupts. Interrupts are generated by events.

A peripheral only occupies one interrupt, and the interrupt number follows the peripheral ID. For example, the peripheral with ID=4 is connected to interrupt number 4 in the nested vectored interrupt controller (NVIC).

Using registers INTEN, INTENSET, and INTENCLR, every event generated by a peripheral can be configured to generate that peripheral's interrupt. Multiple events can be enabled to generate interrupts simultaneously. To resolve the correct interrupt source, the event registers in the event group of peripheral registers will indicate the source.

Some peripherals implement only INTENSET and INTENCLR registers, and the INTEN register is not available on those peripherals. See the individual peripheral chapters for details. In all cases, reading back the INTENSET or INTENCLR register returns the same information as in INTEN.

Each event implemented in the peripheral is associated with a specific bit position in the INTEN, INTENSET and INTENCLR registers.

The relationship between tasks, events, shortcuts, and interrupts is illustrated in figure Tasks, events, shortcuts, publish, subscribe and interrupts on page 15.

Interrupt clearing

Interrupts should always be cleared.

Clearing an interrupt by writing 0 to an event register, or disabling an interrupt using the INTENCLR register, may take a number of CPU clock cycles to take effect. This means that an interrupt may reoccur immediately, even if a new event has not come, if the program exits an interrupt handler after the interrupt is cleared or disabled but before it has taken effect.

> **Note:** To avoid an interrupt reoccurring before a new event has come, the program should perform a read from one of the peripheral registers. For example, the event register that has been cleared, or the INTENCLR register that has been used to disable the interrupt.

Care should be taken to ensure that the compiler does not remove the read operation as an optimization.

3.3.10 Secure/non-secure peripherals

For some peripherals, the security configuration can change from secure to non-secure, or vice versa. Care must be taken when changing the security configuration of a peripheral, to prevent security information leakage and ensure correct operation.

The following sequence should be followed, where applicable, when configuring and changing the security settings of a peripheral in the SPU - System protection unit on page 260:

1. Stop peripheral operation
2. Disable the peripheral
3. Remove pin connections
4. Disable DPPI connections
5. Clear sensitive registers (e.g. writing back default values)
6. Change peripheral security setting in the SPU - System protection unit on page 260
7. Re-enable the peripheral

4 Application core

4.1 CPU

The ARM® Cortex-M33 processor has a 32-bit instruction set (Thumb®-2 technology) that implements a superset of 16 and 32-bit instructions to maximize code density and performance.

This processor implements several features that enable energy-efficient arithmetic and high-performance signal processing, including:

- Digital signal processing (DSP) instructions
- Single-cycle multiply and accumulate (MAC) instructions
- Hardware divide
- 8- and 16-bit single instruction, multiple data (SIMD) instructions
- Single-precision floating-point unit (FPU)
- Memory Protection Unit (MPU)
- ARM® TrustZone® for ARMv8-M

The ARM® Cortex Microcontroller Software Interface Standard (CMSIS) hardware abstraction layer for the ARM® Cortex processor series is implemented and available for the M33 CPU.

Real-time execution is highly deterministic in thread mode, to and from sleep modes, and when handling events at configurable priority levels via the nested vectored interrupt controller (NVIC).

Executing code from internal or external flash will have a wait state penalty. The instruction cache can be enabled to minimize flash wait states when fetching instructions. For more information on cache, see Cache on page 31. The section Electrical specification on page 20 shows CPU performance parameters including the wait states in different modes, CPU current and efficiency, and processing power and efficiency based on the CoreMark® benchmark.

4.1.1 CPU and support module configuration

The ARM® Cortex®-M33 processor has a number of CPU options and support modules implemented on the device.

Option / Module	Description	Implemented
Core options		
NVIC	Nested vectored interrupt controller	
PRIORITIES	Priority bits	3
WIC	Wake-up interrupt controller	NO
Endianness	Memory system endianness	Little endian
DWT	Data watchpoint and trace	YES
Modules		
MPU_NS	Number of non-secure memory protection unit (MPU) regions	16
MPU_S	Number of secure MPU regions	16
SAU	Number of security attribution unit (SAU) regions	0, see SPU for more information about secure regions.
FPU	Floating-point unit	YES
DSP	Digital signal processing extension	YES
ARMv8-M TrustZone®	ARMv8-M security extensions	YES
CPIF	Co-processor interface	NO
ETM	Embedded trace macrocell	YES
ITM	Instrumentation trace macrocell	YES
MTB	Micro trace buffer	NO
CTI	Cross trigger interface	YES
BPU	Breakpoint unit	YES
HTM	AMBA™ AHB trace macrocell	NO

4.1.2 Electrical specification

4.1.2.1 CPU performance

The CPU clock speed is 64 MHz. Current and efficiency data is taken when in System ON and the CPU is executing the CoreMark™ benchmark. It includes power regulator and clock base currents. All other blocks are IDLE.

Symbol	Description	Min.	Typ.	Max.	Units
W_{FLASH}	CPU wait states, running from flash, cache disabled	0		4	
$W_{FLASHCACHE}$	CPU wait states, running from flash, cache enabled	0		2	
W_{RAM}	CPU wait states, running from RAM			0	
CM_{FLASH}	CoreMark[1], running from flash, cache enabled		243		CoreM
$CM_{FLASH/MHz}$	CoreMark per MHz, running from flash, cache enabled		3.79		CoreMark/ MHz
$CM_{FLASH/mA}$	CoreMark per mA, running from flash, cache enabled, DC/ DC		84		CoreM mA

4.2 Memory

The application microcontroller has embedded 1024 kB flash and 256 kB RAM for application code and data storage.

As illustrated in Memory layout on page 21, both CPU and EasyDMA are able to access RAM via the AHB multilayer interconnect. See AHB multilayer interconnect on page 47 and EasyDMA on page 44 for more information about AHB multilayer interconnect and EasyDMA respectively. The LTE modem can access all application MCU memory, but typically a small portion of RAM is dedicated to data exchange between application MCU and the modem baseband controller.

[1] Using IAR compiler

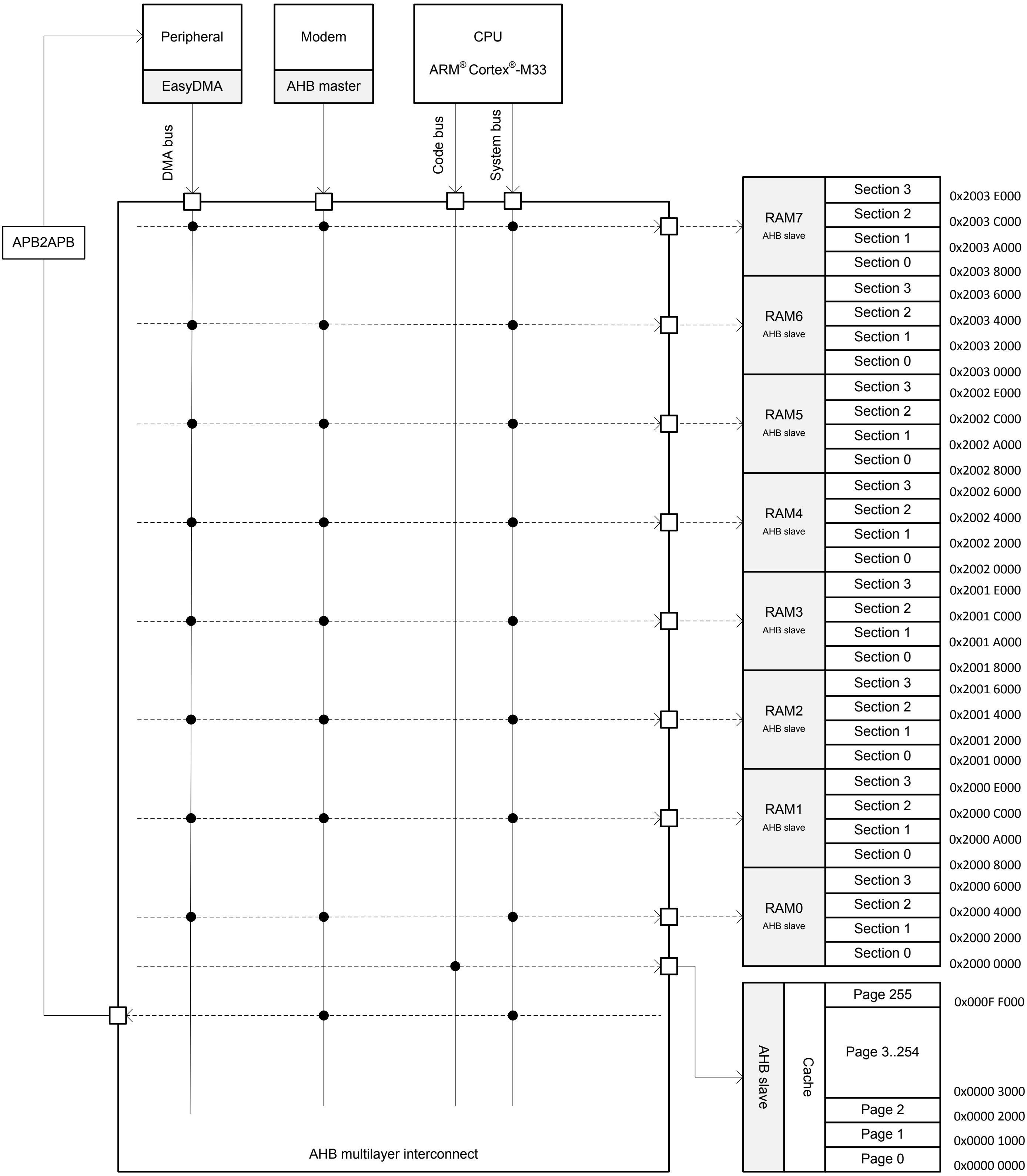

Figure 3: Memory layout

RAM - Random access memory

RAM can be read and written an unlimited number of times by the CPU and the EasyDMA.

Each RAM AHB slave is connected to one or more RAM sections. See Memory layout on page 21 for more information.

The RAM blocks power states and retention states in System ON and System OFF modes are controlled by the VMC.

Flash - Non-volatile memory

Flash can be read an unlimited number of times by the CPU and is accessible via the AHB interface connected to the CPU, see Memory layout on page 21 for more information. There are restrictions on the number of times flash can be written and erased, and also on how it can be written. For more information, see Absolute maximum ratings on page 392. Writing to flash is managed by the non-volatile memory controller (NVMC).

4.2.1 Memory map

All memory and registers are found in the same address space, as illustrated in the device memory map below.

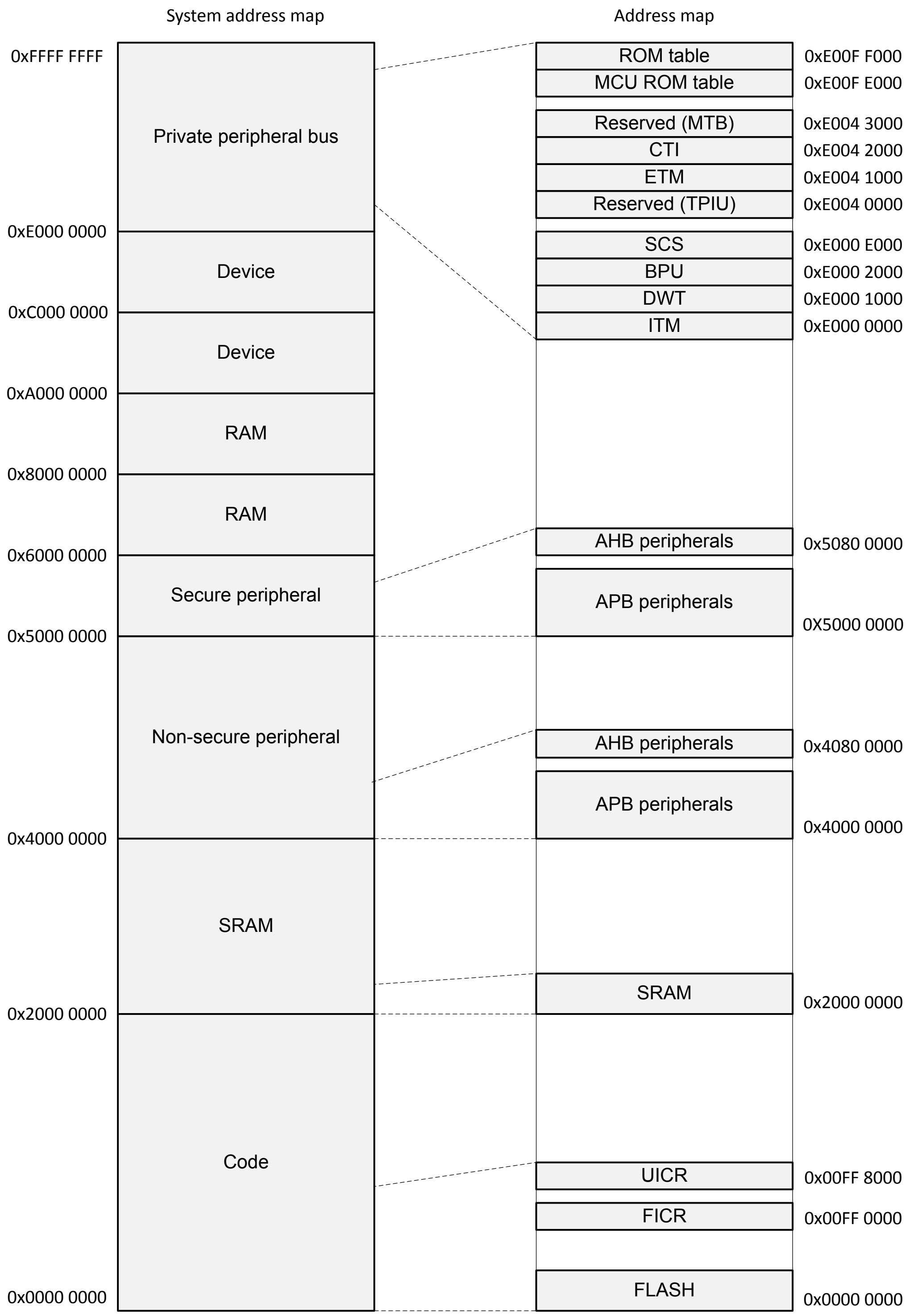

Figure 4: Memory map

Some of the registers are retained (their values kept). Read more about retained registers in Retained registers on page 55 and Reset behavior on page 55.

4.2.2 Instantiation

ID	Base address	Peripheral	Instance	Secure mapping	DMA security	Description
3	0x50003000	SPU	SPU	S	NA	System Protection Unit

ID	Base address	Peripheral	Instance	Secure mapping	DMA security	Description
4	0x50004000 0x40004000	REGULATORS	REGULATORS : S REGULATORS : NS	US	NA	Regulator configuration
5	0x50005000 0x40005000	CLOCK	CLOCK : S CLOCK : NS	US	NA	Clock control
5	0x50005000 0x40005000	POWER	POWER : S POWER : NS	US	NA	Power control
6	0x50006000	CTRLAPPERI	CTRL_AP_PERI	S	NA	CTRL-AP-PERI
8	0x50008000 0x40008000	SPIM	SPIM0 : S SPIM0 : NS	US	SA	SPI master 0
8	0x50008000 0x40008000	SPIS	SPIS0 : S SPIS0 : NS	US	SA	SPI slave 0
8	0x50008000 0x40008000	TWIM	TWIM0 : S TWIM0 : NS	US	SA	Two-wire interface master 0
8	0x50008000 0x40008000	TWIS	TWIS0 : S TWIS0 : NS	US	SA	Two-wire interface slave 0
8	0x50008000 0x40008000	UARTE	UARTE0 : S UARTE0 : NS	US	SA	Universal asynchronous receiver/transmitter with EasyDMA 0
9	0x50009000 0x40009000	SPIM	SPIM1 : S SPIM1 : NS	US	SA	SPI master 1
9	0x50009000 0x40009000	SPIS	SPIS1 : S SPIS1 : NS	US	SA	SPI slave 1
9	0x50009000 0x40009000	TWIM	TWIM1 : S TWIM1 : NS	US	SA	Two-wire interface master 1
9	0x50009000 0x40009000	TWIS	TWIS1 : S TWIS1 : NS	US	SA	Two-wire interface slave 1
9	0x50009000 0x40009000	UARTE	UARTE1 : S UARTE1 : NS	US	SA	Universal asynchronous receiver/transmitter with EasyDMA 1
10	0x5000A000 0x4000A000	SPIM	SPIM2 : S SPIM2 : NS	US	SA	SPI master 2
10	0x5000A000 0x4000A000	SPIS	SPIS2 : S SPIS2 : NS	US	SA	SPI slave 2
10	0x5000A000 0x4000A000	TWIM	TWIM2 : S TWIM2 : NS	US	SA	Two-wire interface master 2
10	0x5000A000 0x4000A000	TWIS	TWIS2 : S TWIS2 : NS	US	SA	Two-wire interface slave 2
10	0x5000A000 0x4000A000	UARTE	UARTE2 : S UARTE2 : NS	US	SA	Universal asynchronous receiver/transmitter with EasyDMA 2
11	0x5000B000 0x4000B000	SPIM	SPIM3 : S SPIM3 : NS	US	SA	SPI master 3
11	0x5000B000 0x4000B000	SPIS	SPIS3 : S SPIS3 : NS	US	SA	SPI slave 3
11	0x5000B000 0x4000B000	TWIM	TWIM3 : S TWIM3 : NS	US	SA	Two-wire interface master 3
11	0x5000B000 0x4000B000	TWIS	TWIS3 : S TWIS3 : NS	US	SA	Two-wire interface slave 3
11	0x5000B000 0x4000B000	UARTE	UARTE3 : S UARTE3 : NS	US	SA	Universal asynchronous receiver/transmitter with EasyDMA 3
13	0x5000D000	GPIOTE	GPIOTE0	S	NA	Secure GPIO tasks and events
14	0x5000E000 0x4000E000	SAADC	SAADC : S SAADC : NS	US	SA	Analog to digital converter
15	0x5000F000 0x4000F000	TIMER	TIMER0 : S TIMER0 : NS	US	NA	Timer 0

ID	Base address	Peripheral	Instance	Secure mapping	DMA security	Description
16	0x50010000 0x40010000	TIMER	TIMER1 : S TIMER1 : NS	US	NA	Timer 1
17	0x50011000 0x40011000	TIMER	TIMER2 : S TIMER2 : NS	US	NA	Timer 2
20	0x50014000 0x40014000	RTC	RTC0 : S RTC0 : NS	US	NA	Real time counter 0
21	0x50015000 0x40015000	RTC	RTC1 : S RTC1 : NS	US	NA	Real time counter 1
23	0x50017000 0x40017000	DPPIC	DPPIC : S DPPIC : NS	SPLIT	NA	DPPI configuration
24	0x50018000 0x40018000	WDT	WDT : S WDT : NS	US	NA	Watchdog timer
27	0x5001B000 0x4001B000	EGU	EGU0 : S EGU0 : NS	US	NA	Event generator unit 0
28	0x5001C000 0x4001C000	EGU	EGU1 : S EGU1 : NS	US	NA	Event generator unit 1
29	0x5001D000 0x4001D000	EGU	EGU2 : S EGU2 : NS	US	NA	Event generator unit 2
30	0x5001E000 0x4001E000	EGU	EGU3 : S EGU3 : NS	US	NA	Event generator unit 3
31	0x5001F000 0x4001F000	EGU	EGU4 : S EGU4 : NS	US	NA	Event generator unit 4
32	0x50020000 0x40020000	EGU	EGU5 : S EGU5 : NS	US	NA	Event generator unit 5
33	0x50021000 0x40021000	PWM	PWM0 : S PWM0 : NS	US	SA	Pulse width modulation unit 0
34	0x50022000 0x40022000	PWM	PWM1 : S PWM1 : NS	US	SA	Pulse width modulation unit 1
35	0x50023000 0x40023000	PWM	PWM2 : S PWM2 : NS	US	SA	Pulse width modulation unit 2
36	0x50024000 0x40024000	PWM	PWM3 : S PWM3 : NS	US	SA	Pulse width modulation unit 3
38	0x50026000 0x40026000	PDM	PDM : S PDM : NS	US	SA	Pulse density modulation (digital microphone) interface
40	0x50028000 0x40028000	I2S	I2S : S I2S : NS	US	SA	Inter-IC Sound
42	0x5002A000 0x4002A000	IPC	IPC : S IPC : NS	US	NA	Interprocessor communication
44	0x5002C000 0x4002C000	FPU	FPU : S FPU : NS	US	NA	Floating-point unit
49	0x40031000	GPIOTE	GPIOTE1	NS	NA	Non Secure GPIO tasks and events
57	0x50039000 0x40039000	KMU	KMU : S KMU : NS	SPLIT	NA	Key management unit
57	0x50039000 0x40039000	NVMC	NVMC : S NVMC : NS	SPLIT	NA	Non-volatile memory controller
58	0x5003A000 0x4003A000	VMC	VMC : S VMC : NS	US	NA	Volatile memory controller
64	0x50840000	CC_HOST_RGF	CC_HOST_RGF	S	NSA	Host platform interface
64	0x50840000	CRYPTOCELL	CRYPTOCELL	S	NSA	CryptoCell sub-system control interface
66	0x50842500 0x40842500	GPIO	P0 : S P0 : NS	SPLIT	NA	General purpose input and output
N/A	0x00FF0000	FICR	FICR	S	NA	Factory information configuration
N/A	0x00FF8000	UICR	UICR	S	NA	User information configuration

ID	Base address	Peripheral	Instance	Secure mapping	DMA security	Description
N/A	0xE0080000	TAD	TAD	S	NA	Trace and debug control

Table 4: Instantiation table

4.2.3 Peripheral access control capabilities

Information about the peripheral access control capabilities can be found in the instantiation table.

The instantiation table has two columns containing the information about access control capabilities for a peripheral:

- Secure mapping: This column defines configuration capabilities for TrustZone®-M secure attribute.
- DMA security: This column indicates if the peripheral has DMA capabilities, and if DMA transfer can be assigned to a different security attribute than the peripheral itself.

For details on options in secure mapping column and DMA security column, see the following tables respecitvely.

Abbreviation	Description
NS	Non-secure: This peripheral is always accessible as a non-secure peripheral.
S	Secure: This peripheral is always accessible as a secure peripheral.
US	User-selectable: Non-secure or secure attribute for this peripheral is defined by the PERIPHID[0].PERM register.
SPLIT	Both non-secure and secure: The same resource is shared by both secure and non-secure code.

Table 5: Secure mapping column options

Abbreviation	Description
NA	Not applicable: Peripheral has no DMA capability.
NSA	No separate attribute: Peripheral has DMA, and DMA transfers always have the same security attribute as assigned to the peripheral.
SA	Separate attribute: Peripheral has DMA, and DMA transfers can have a different security attribute than the one assigned to the peripheral.

Table 6: DMA security column options

4.3 VMC — Volatile memory controller

The volatile memory controller (VMC) provides power control of RAM blocks.

Each of the available RAM blocks, which can contain multiple RAM sections, can be turned on or off independently in System ON mode, using the RAM[n]registers. These registers also control if a RAM block, or some of its sections, is retained in System OFF mode. See Memory chapter for more information about RAM blocks and sections.

> **Note:** Powering up a RAM block takes typically 10 cycles. Thus, it is recommended reading the POWER register before accessing a RAM block that has been recently powered on.

4.3.1 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x5003A000 0x4003A000	VMC	VMC : S VMC : NS	US	NA	Volatile memory controller	

Table 7: Instances

Register	Offset	Security	Description
RAM[0].POWER	0x600		RAM0 power control register
RAM[0].POWERSET	0x604		RAM0 power control set register
RAM[0].POWERCLR	0x608		RAM0 power control clear register
RAM[1].POWER	0x610		RAM1 power control register
RAM[1].POWERSET	0x614		RAM1 power control set register
RAM[1].POWERCLR	0x618		RAM1 power control clear register
RAM[2].POWER	0x620		RAM2 power control register
RAM[2].POWERSET	0x624		RAM2 power control set register
RAM[2].POWERCLR	0x628		RAM2 power control clear register
RAM[3].POWER	0x630		RAM3 power control register
RAM[3].POWERSET	0x634		RAM3 power control set register
RAM[3].POWERCLR	0x638		RAM3 power control clear register
RAM[4].POWER	0x640		RAM4 power control register
RAM[4].POWERSET	0x644		RAM4 power control set register
RAM[4].POWERCLR	0x648		RAM4 power control clear register
RAM[5].POWER	0x650		RAM5 power control register
RAM[5].POWERSET	0x654		RAM5 power control set register
RAM[5].POWERCLR	0x658		RAM5 power control clear register
RAM[6].POWER	0x660		RAM6 power control register
RAM[6].POWERSET	0x664		RAM6 power control set register
RAM[6].POWERCLR	0x668		RAM6 power control clear register
RAM[7].POWER	0x670		RAM7 power control register
RAM[7].POWERSET	0x674		RAM7 power control set register
RAM[7].POWERCLR	0x678		RAM7 power control clear register

Table 8: Register overview

4.3.1.1 RAM[n].POWER (n=0..7)

Address offset: 0x600 + (n × 0x10)

RAMn power control register

Bit number	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
ID	H G F E (bits 19–16) D C B A (bits 3–0)
Reset 0x0000FFFF	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

ID	Acce	Field	Value ID	Value	Description
A-D	RW	S[i]POWER (i=0..3)			Keep RAM section Si of RAM n on or off in System ON mode All RAM sections will be switched off in System OFF mode
			Off	0	Off
			On	1	On
E-H	RW	S[i]RETENTION (i=0..3)			Keep retention on RAM section Si of RAM n when RAM section is switched off
			Off	0	Off
			On	1	On

4.3.1.2 RAM[n].POWERSET (n=0..7)

Address offset: 0x604 + (n × 0x10)

RAMn power control set register

When read, this register will return the value of the POWER register.

Bit number	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
ID	H G F E (bits 19–16) D C B A (bits 3–0)
Reset 0x0000FFFF	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

ID	Acce	Field	Value ID	Value	Description
A-D	W	S[i]POWER (i=0..3)			Keep RAM section Si of RAM n on or off in System ON mode
			On	1	On
E-H	W	S[i]RETENTION (i=0..3)			Keep retention on RAM section Si of RAM n when RAM section is switched off
			On	1	On

4.3.1.3 RAM[n].POWERCLR (n=0..7)

Address offset: 0x608 + (n × 0x10)

RAMn power control clear register

When read, this register will return the value of the POWER register.

Bit number	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
ID	H G F E (bits 19–16) D C B A (bits 3–0)
Reset 0x0000FFFF	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

ID	Acce	Field	Value ID	Value	Description
A-D	W	S[i]POWER (i=0..3)			Keep RAM section Si of RAM n on or off in System ON mode
			Off	1	Off
E-H	W	S[i]RETENTION (i=0..3)			Keep retention on RAM section Si of RAM n when RAM section is switched off
			Off	1	Off

4.4 NVMC — Non-volatile memory controller

The non-volatile memory controller (NVMC) is used for writing and erasing of the internal flash memory and the user information configuration register (UICR).

The NVMC is a split security peripheral. This means that when the NVMC is configured as non-secure, only a subset of the registers is available from the non-secure code. See SPU - System protection unit on page 260 and Registers on page 31 for more details.

When the NVMC is configured to be a secure peripheral, only secure code has access.

Before a write can be performed, the NVMC must be enabled for writing in CONFIG.WEN. Similarly, before an erase can be performed, the NVMC must be enabled for erasing in CONFIG.EEN, see CONFIG on page 32. The user must make sure that writing and erasing are not enabled at the same time. Failing to do so may result in unpredictable behavior.

4.4.1 Writing to flash

When writing is enabled, in CONFIG register for secure region, or in CONFIGNS register for non-secure region, flash is written by writing a full 32-bit word to a word-aligned address in flash.

Secure code has access to both secure and non-secure regions, by using the appropriate configuration of CONFIG and CONFIGNS registers. Non-secure code, in constrast, has access to non-secure regions only. Thus, non-secure code only needs CONFIGNS.

The NVMC is only able to write '0' to erased bits in flash, that is bits set to '1'. It cannot write a bit back to '1'.

As illustrated in Memory on page 20, flash is divided into multiple pages. The same address in flash can only be written n_{WRITE} number of times before a page erase must be performed.

Only full 32-bit words can be written to flash using the NVMC interface. To write less than 32 bits to flash, write the data as a word, and set all the bits that should remain unchanged in the word to '1'. Note that the restriction about the number of writes (see above) still applies in this case.

The time it takes to write a word to flash is specified by t_{WRITE}. If CPU executes code from flash while the NVMC is writing to flash, the CPU will be stalled.

Only word-aligned writes are allowed. Byte or half-word-aligned writes will result in a bus fault.

4.4.2 Erasing a secure page in flash

When secure region erase is enabled (in CONFIG register), a flash page can be erased by writing 0xFFFFFFFF into the first 32-bit word in a flash page.

Page erase is only applicable to the code area in the flash and does not work with UICR.

After erasing a flash page, all bits in the page are set to '1'. The time it takes to erase a page is specified by $t_{ERASEPAGE}$. The CPU is stalled if the CPU executes code from the flash while the NVMC performs the erase operation.

See Partial erase of a page in flash for information on splitting the erase time in smaller chunks.

4.4.3 Erasing a non-secure page in flash

When non-secure region erase is enabled, a non-secure flash page can be erased by writing 0xFFFFFFFF into the first 32-bit word of the flash page.

Page erase is only applicable to the code area in the flash and does not work with UICR.

After erasing a flash page, all bits in the page are set to '1'. The time it takes to erase a page is specified by $t_{ERASEPAGE}$. The CPU is stalled if the CPU executes code from the flash while the NVMC performs the erase operation.

4.4.4 Writing to user information configuration registers (UICR)

User information configuration registers (UICR) are written in the same way as flash. After UICR has been written, the new UICR configuration will only take effect after a reset.

UICR is only accessible by secure code. Any write from non-secure code will be faulted. In order to lock the chip after uploading non-secure code, non-secure debugger needs to use the WRITEUICRNS register inside the NVMC in order to set APPROTECT (APPROTECT will be written to 0x00000000).

UICR can only be written n_{WRITE} number of times before an erase must be performed using ERASEALL.

The time it takes to write a word to the UICR is specified by t_{WRITE}. The CPU is stalled if the CPU executes code from the flash while the NVMC is writing to the UICR.

4.4.5 Erase all

When erase is enabled, the whole flash and UICR can be erased in one operation by using the ERASEALL register. ERASEALL will not erase the factory information configuration registers (FICR).

This functionality can be blocked by some configuration of the UICR protection bits, see the table NVMC protection (1 - Enabled, 0 - Disabled, X - Don't care) on page 30.

The time it takes to perform an ERASEALL on page 32 command is specified by $t_{ERASEALL}$. The CPU is stalled if the CPU executes code from the flash while the NVMC performs the erase operation.

4.4.6 NVMC protection mechanisms

This chapter describes the different protection mechanisms for the non-volatile memory.

4.4.6.1 NVMC blocking

UICR integrity is assured through use of multiple levels of protection. UICR protection bits can be configured to allow or block certain operations.

The table below shows the different status of UICR protection bits, and which operations are allowed or blocked.

UICR protection bit status			NVMC protection	
SECUREAPPROTECT	APPROTECT	ERASEPROTECT	CTRL-AP ERASEALL	NVMC ERASEALL
0	0	0	Available	Available
1	X	0	Available	Blocked
X	1	0	Available	Blocked
X	X	1	Blocked	Blocked

Table 9: NVMC protection (1 - Enabled, 0 - Disabled, X - Don't care)

> **Note:** Erase can still be performed through CTRL-AP, regardless of the above settings. See CTRL-AP - Control access port on page 374 for more information.

Uploading code with secure debugging blocked

Non-secure code can program non-secure flash regions. In order to perform these operations, the NVMC has the following non-secure registers: CONFIGNS, READY and READYNEXT.

Register CONFIGNS on page 34 works as the CONFIG register but it is used only for non-secure transactions. Both page erase and writing inside the flash require a write transaction (see Erasing a secure page in flash on page 29 or Erasing a non-secure page in flash on page 29). Because of this, the SPU - System protection unit on page 260 will guarantee that the non-secure code cannot write inside a secure page, since the transaction will never reach the NVMC controller.

4.4.6.2 NVMC power failure protection

NVMC power failure protection is possible through use of power-fail comparator that is monitoring power supply.

If the power-fail comparator is enabled, and the power supply voltage is below V_{POF} threshold, the power-fail comparator will prevent the NVMC from performing erase or write operations in non-volatile memory (NVM).

If a power failure warning is present at the start of an NVM write or erase operation, the NVMC will block the operation and a bus error will be signalled. If a power failure warning occurs during an ongoing NVM write operation, the NVMC will try to finish the operation. And if the power failure warning persists, consecutive NVM write operations will be blocked by the NVMC, and a bus error will be signalled. If a power failure warning occurs during an NVM erase operation, the operation is aborted and a bus error is signalled.

4.4.7 Cache

An instruction cache (I-Cache) can be enabled for the ICODE bus in the NVMC.

See Memory map on page 22 for the location of flash.

A cache hit is an instruction fetch from the cache, and it has a 0 wait-state delay. The number of wait-states for a cache miss, where the instruction is not available in the cache and needs to be fetched from flash, depends on the processor frequency and is shown in CPU on page 19.

Enabling the cache can increase the CPU performance, and reduce power consumption by reducing the number of wait cycles and the number of flash accesses. This will depend on the cache hit rate. Cache draws current when enabled. If the reduction in average current due to reduced flash accesses is larger than the cache power requirement, the average current to execute the program code will be reduced.

When disabled, the cache does not draw current and its content is not retained.

It is possible to enable cache profiling to analyze the performance of the cache for your program using the register ICACHECNF. When profiling is enabled, registers IHIT and IMISS are incremented for every instruction cache hit or miss respectively.

4.4.8 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50039000 0x40039000	NVMC	NVMC : S NVMC : NS	SPLIT	NA	Non-volatile memory controller	

Table 10: Instances

Register	Offset	Security	Description
READY	0x400	NS	Ready flag
READYNEXT	0x408	NS	Ready flag
CONFIG	0x504	S	Configuration register
ERASEALL	0x50C	S	Register for erasing all non-volatile user memory
ERASEPAGEPARTIALCFG	0x51C	S	Register for partial erase configuration
ICACHECNF	0x540	S	I-code cache configuration register
IHIT	0x548	S	I-code cache hit counter
IMISS	0x54C	S	I-code cache miss counter
CONFIGNS	0x584	NS	
WRITEUICRNS	0x588	NS	Non-secure APPROTECT enable register

Table 11: Register overview

4.4.8.1 READY

Address offset: 0x400

Ready flag

Bit number	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
ID																																A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	READY			NVMC is ready or busy
			Busy	0	NVMC is busy (on-going write or erase operation)
			Ready	1	NVMC is ready

4.4.8.2 READYNEXT

Address offset: 0x408

Ready flag

Bit number	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
ID																																A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	READYNEXT			NVMC can accept a new write operation
			Busy	0	NVMC cannot accept any write operation
			Ready	1	NVMC is ready

4.4.8.3 CONFIG

Address offset: 0x504

Configuration register

This register is one hot

Bit number	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
ID																														A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	WEN			Program memory access mode. It is strongly recommended to only activate erase and write modes when they are actively used. Enabling write or erase will invalidate the cache and keep it invalidated.
			Ren	0	Read only access
			Wen	1	Write enabled
			Een	2	Erase enabled
			PEen	4	Partial erase enabled

4.4.8.4 ERASEALL

Address offset: 0x50C

Register for erasing all non-volatile user memory

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	ERASEALL			Erase all non-volatile memory including UICR registers. Note that erasing must be enabled by setting CONFIG.WEN = Een before the non-volatile memory can be erased.
			NoOperation	0	No operation
			Erase	1	Start chip erase

4.4.8.5 ERASEPAGEPARTIALCFG

Address offset: 0x51C

Register for partial erase configuration

Bit number	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
ID																										A	A	A	A	A	A	A
Reset 0x0000000A	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	DURATION			Duration of the partial erase in milliseconds The user must ensure that the total erase time is long enough for a complete erase of the flash page

4.4.8.6 ICACHECNF

Address offset: 0x540

I-code cache configuration register

Bit number	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
ID																								B								A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CACHEEN			Cache enable
			Disabled	0	Disable cache. Invalidates all cache entries.
			Enabled	1	Enable cache
B	RW	CACHEPROFEN			Cache profiling enable
			Disabled	0	Disable cache profiling
			Enabled	1	Enable cache profiling

4.4.8.7 IHIT

Address offset: 0x548

I-code cache hit counter

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	HITS			Number of cache hits Write zero to clear

4.4.8.8 IMISS

Address offset: 0x54C

I-code cache miss counter

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MISSES			Number of cache misses Write zero to clear

4.4.8.9 CONFIGNS

Address offset: 0x584

This register is one hot

Bit number	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
ID																															A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	WEN			Program memory access mode. It is strongly recommended to only activate erase and write modes when they are actively used. Enabling write or erase will invalidate the cache and keep it invalidated.
			Ren	0	Read only access
			Wen	1	Write enabled
			Een	2	Erase enabled

4.4.8.10 WRITEUICRNS

Address offset: 0x588

Non-secure APPROTECT enable register

Bit number	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
ID	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B				A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	SET			Allow non-secure code to set APPROTECT
			Set	1	Set value
B	W	KEY			Key to write in order to validate the write operation
			Keyvalid	0xAFBE5A7	Key value

4.4.9 Electrical specification

4.4.9.1 Flash programming

Symbol	Description	Min.	Typ.	Max.	Units
n_{WRITE}	Number of times a 32-bit word can be written before erase			2	
$n_{ENDURANCE}$	Erase cycles per page	10,000			
t_{WRITE}	Time to write one 32-bit word			43	µs
$t_{ERASEPAGE}$	Time to erase one page			87	ms
$t_{ERASEALL}$	Time to erase all flash			173	ms
$t_{ERASEPAGEPARTIAL,setup}$	Setup time for one partial erase			1.08	ms

4.4.9.2 Cache size

Symbol	Description	Min.	Typ.	Max.	Units
$Size_{ICODE}$	I-Code cache size		2048		Bytes

4.5 FICR — Factory information configuration registers

Factory information configuration registers (FICR) are pre-programmed in factory and cannot be erased by the user. These registers contain chip-specific information and configuration.

4.5.1 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x00FF0000	FICR	FICR	S	NA	Factory information configuration	

Table 12: Instances

Register	Offset	Security	Description
INFO.DEVICEID[0]	0x204		Device identifier
INFO.DEVICEID[1]	0x208		Device identifier
INFO.PART	0x20C		Part code
INFO.VARIANT	0x210		Part Variant, Hardware version and Production configuration
INFO.PACKAGE	0x214		Package option
INFO.RAM	0x218		RAM variant
INFO.FLASH	0x21C		Flash variant
INFO.CODEPAGESIZE	0x220		Code memory page size
INFO.CODESIZE	0x224		Code memory size
INFO.DEVICETYPE	0x228		Device type
TRIMCNF[n].ADDR	0x300		Address
TRIMCNF[n].DATA	0x304		Data
TRNG90B.BYTES	0xC00		Amount of bytes for the required entropy bits
TRNG90B.RCCUTOFF	0xC04		Repetition counter cutoff
TRNG90B.APCUTOFF	0xC08		Adaptive proportion cutoff
TRNG90B.STARTUP	0xC0C		Amount of bytes for the startup tests
TRNG90B.ROSC1	0xC10		Sample count for ring oscillator 1
TRNG90B.ROSC2	0xC14		Sample count for ring oscillator 2
TRNG90B.ROSC3	0xC18		Sample count for ring oscillator 3

Register	Offset	Security	Description
TRNG90B.ROSC4	0xC1C		Sample count for ring oscillator 4

Table 13: Register overview

4.5.1.1 INFO.DEVICEID[n] (n=0..1)

Address offset: 0x204 + (n × 0x4)

Device identifier

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	DEVICEID			64 bit unique device identifier DEVICEID[0] contains the least significant bits of the device identifier. DEVICEID[1] contains the most significant bits of the device identifier.

4.5.1.2 INFO.PART

Address offset: 0x20C

Part code

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	PART			Part code
			N9160	0x9160	nRF9160

4.5.1.3 INFO.VARIANT

Address offset: 0x210

Part Variant, Hardware version and Production configuration

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x0FFFFFFF	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	VARIANT			Part Variant, Hardware version and Production configuration, encoded as ASCII
			AAAA	0x41414141	AAAA
			AAA0	0x41414130	AAA0

4.5.1.4 INFO.PACKAGE

Address offset: 0x214

Package option

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00002000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	PACKAGE			Package option
			CC	0x2000	CCxx - 236 ball wlCSP

4.5.1.5 INFO.RAM

Address offset: 0x218

RAM variant

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000100	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	RAM			RAM variant
			K256	0x100	256 kByte RAM
			Unspecified	0xFFFFFFFF	Unspecified

4.5.1.6 INFO.FLASH

Address offset: 0x21C

Flash variant

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000400	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	FLASH			Flash variant
			K1024	0x400	1 MByte FLASH

4.5.1.7 INFO.CODEPAGESIZE

Address offset: 0x220

Code memory page size

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00001000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	CODEPAGESIZE			Code memory page size

4.5.1.8 INFO.CODESIZE

Address offset: 0x224

Code memory size

Bit number	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	
ID	A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A	
Reset 0x00000100	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0**	

ID	Acce	Field	Value ID	Value	Description
A	R	CODESIZE			Code memory size in number of pages Total code space is: CODEPAGESIZE * CODESIZE

4.5.1.9 INFO.DEVICETYPE

Address offset: 0x228

Device type

Bit number	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
ID	A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
Reset 0xFFFFFFFF	**1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

ID	Acce	Field	Value ID	Value	Description
A	R	DEVICETYPE			Device type
			Die	0x0000000	Device is an physical DIE
			FPGA	0xFFFFFFFF	Device is an FPGA

4.5.1.10 TRIMCNF[n].ADDR (n=0..255)

Address offset: 0x300 + (n × 0x8)

Address

Bit number	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
ID	A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
Reset 0xFFFFFFFF	**1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

ID	Acce	Field	Value ID	Value	Description
A	R	Address			Address

4.5.1.11 TRIMCNF[n].DATA (n=0..255)

Address offset: 0x304 + (n × 0x8)

Data

Bit number	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
ID	A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
Reset 0xFFFFFFFF	**1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

ID	Acce	Field	Value ID	Value	Description
A	R	Data			Data

4.5.1.12 TRNG90B.BYTES

Address offset: 0xC00

Amount of bytes for the required entropy bits

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	BYTES			Amount of bytes for the required entropy bits

4.5.1.13 TRNG90B.RCCUTOFF

Address offset: 0xC04

Repetition counter cutoff

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	RCCUTOFF			Repetition counter cutoff

4.5.1.14 TRNG90B.APCUTOFF

Address offset: 0xC08

Adaptive proportion cutoff

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	APCUTOFF			Adaptive proportion cutoff

4.5.1.15 TRNG90B.STARTUP

Address offset: 0xC0C

Amount of bytes for the startup tests

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000210	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	STARTUP			Amount of bytes for the startup tests

4.5.1.16 TRNG90B.ROSC1

Address offset: 0xC10

Sample count for ring oscillator 1

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	ROSC1			Sample count for ring oscillator 1

4.5.1.17 TRNG90B.ROSC2

Address offset: 0xC14

Sample count for ring oscillator 2

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	ROSC2			Sample count for ring oscillator 2

4.5.1.18 TRNG90B.ROSC3

Address offset: 0xC18

Sample count for ring oscillator 3

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	ROSC3			Sample count for ring oscillator 3

4.5.1.19 TRNG90B.ROSC4

Address offset: 0xC1C

Sample count for ring oscillator 4

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	ROSC4			Sample count for ring oscillator 4

4.6 UICR — User information configuration registers

The user information configuration registers (UICRs) are non-volatile memory (NVM) registers for configuring user specific settings.

For information on writing UICR registers, see the NVMC — Non-volatile memory controller on page 28 and Memory on page 20 chapters.

4.6.1 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x00FF8000	UICR	UICR	S	NA	User information configuration	

Table 14: Instances

Register	Offset	Security	Description
APPROTECT	0x000		Access port protection
UNUSED0	0x004		Reserved
UNUSED1	0x008		Reserved
UNUSED2	0x00C		Reserved
UNUSED3	0x010		Reserved
XOSC32M	0x014		Oscillator control
HFXOSRC	0x01C		HFXO clock source selection
HFXOCNT	0x020		HFXO startup counter
SECUREAPPROTECT	0x02C		Secure access port protection
ERASEPROTECT	0x030		Erase protection
OTP[n]	0x108		One time programmable memory
KEYSLOT.CONFIG[n].DEST	0x400		Destination address where content of the key value registers (KEYSLOT.KEYn.VALUE[0-3]) will be pushed by KMU. Note that this address MUST match that of a peripherals APB mapped write-only key registers, else the KMU can push this key value into an address range which the CPU can potentially read!
KEYSLOT.CONFIG[n].PERM	0x404		Define permissions for the key slot. Bits 0-15 and 16-31 can only be written when equal to 0xFFFF.
KEYSLOT.KEY[n].VALUE[0]	0x800		Define bits [31:0] of value assigned to KMU key slot.
KEYSLOT.KEY[n].VALUE[1]	0x804		Define bits [63:32] of value assigned to KMU key slot.
KEYSLOT.KEY[n].VALUE[2]	0x808		Define bits [95:64] of value assigned to KMU key slot.
KEYSLOT.KEY[n].VALUE[3]	0x80C		Define bits [127:96] of value assigned to KMU key slot.

Table 15: Register overview

4.6.1.1 APPROTECT

Address offset: 0x000

Access port protection

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PALL			Blocks debugger read/write access to all CPU registers and memory mapped addresses
			Unprotected	0xFFFFFFFF	Unprotected
			Protected	0x00000000	Protected

4.6.1.2 XOSC32M

Address offset: 0x014

Oscillator control

Bit number	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
ID																											A	A	A	A	A	A
Reset 0xFFFFFFCF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	CTRL			Pierce current DAC control signals

4.6.1.3 HFXOSRC

Address offset: 0x01C

HFXO clock source selection

Bit number	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
ID																																A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	HFXOSRC			HFXO clock source selection
			XTAL	1	32 MHz crystal oscillator
			TCXO	0	32 MHz temperature compensated crystal oscillator (TCXO)

4.6.1.4 HFXOCNT

Address offset: 0x020

HFXO startup counter

Bit number	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
ID																									A	A	A	A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	HFXOCNT			HFXO startup counter. Total debounce time = HFXOCNT*64 us + 0.5 us
			MinDebounceTime	0	Min debounce time = (0*64 us + 0.5 us)
			MaxDebounceTime	255	Max debounce time = (255*64 us + 0.5 us)

4.6.1.5 SECUREAPPROTECT

Address offset: 0x02C

Secure access port protection

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PALL			Blocks debugger read/write access to all secure CPU registers and secure memory mapped addresses
			Unprotected	0xFFFFFFFF	Unprotected
			Protected	0x00000000	Protected

4.6.1.6 ERASEPROTECT

Address offset: 0x030

Erase protection

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PALL			Blocks NVMC ERASEALL and CTRLAP ERASEALL functionality
			Unprotected	0xFFFFFFFF	Unprotected
			Protected	0x00000000	Protected

4.6.1.7 OTP[n] (n=0..189)

Address offset: 0x108 + (n × 0x4)

One time programmable memory

Bit number	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
ID	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW1	LOWER			Lower half word **Note:** Can only be written to a non 0xFFFF value once.
B	RW1	UPPER			Upper half word **Note:** Can only be written to a non 0xFFFF value once.

4.6.1.8 KEYSLOT.CONFIG[n].DEST (n=0..127)

Address offset: 0x400 + (n × 0x8)

Destination address where content of the key value registers (KEYSLOT.KEYn.VALUE[0-3]) will be pushed by KMU. Note that this address MUST match that of a peripherals APB mapped write-only key registers, else the KMU can push this key value into an address range which the CPU can potentially read!

Writing/reading this register requires the KMU SELECTKEYSLOT register to be set to n+1.

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	DEST			Secure APB destination address

4.6.1.9 KEYSLOT.CONFIG[n].PERM (n=0..127)

Address offset: 0x404 + (n × 0x8)

Define permissions for the key slot. Bits 0-15 and 16-31 can only be written when equal to 0xFFFF.

Writing/reading this register requires the KMU SELECTKEYSLOT register to be set to n+1.

Bit number	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
ID																D														C	B	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	WRITE			Write permission for key slot
			Disabled	0	Disable write to the key value registers
			Enabled	1	Enable write to the key value registers
B	RW	READ			Read permission for key slot
			Disabled	0	Disable read from key value registers
			Enabled	1	Enable read from key value registers
C	RW	PUSH			Push permission for key slot

Bit number				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
ID				D D D D D D D D D D D D D D D D D D D D D D D D D D D D D C B A
Reset 0xFFFFFFFF				**1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**
ID	Acce	Field	Value ID	Value	Description
			Disabled	0	Disable pushing of key value registers over secure APB, but can be read if field READ is Enabled
			Enabled	1	Enable pushing of key value registers over secure APB. Register KEYSLOT.CONFIGn.DEST must contain a valid destination address!
D	RW	STATE			Revocation state for the key slot Note that it is not possible to undo a key revocation by writing the value '1' to this field
			Revoked	0	Key value registers can no longer be read or pushed
			Active	1	Key value registers are readable (if enabled) and can be pushed (if enabled)

4.6.1.10 KEYSLOT.KEY[n].VALUE[o] (n=0..127) (o=0..3)

Address offset: 0x800 + (n × 0x10) + (o × 0x4)

Define bits [31+o*32:0+o*32] of value assigned to KMU key slot.

Writing/reading this register requires the KMU SELECTKEYSLOT register to be set to n+1.

Bit number					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
ID					A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
Reset 0xFFFFFFFF					**1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**
ID	Acce	Field	Value ID	Value	Description
A	RW	VALUE			Define bits [31+o*32:0+o*32] of value assigned to KMU key slot

4.7 EasyDMA

EasyDMA is a module implemented by some peripherals to gain direct access to Data RAM.

EasyDMA is an AHB bus master similar to CPU and is connected to the AHB multilayer interconnect for direct access to Data RAM. EasyDMA is not able to access flash.

A peripheral can implement multiple EasyDMA instances to provide dedicated channels. For example, for reading and writing of data between the peripheral and RAM. This concept is illustrated in EasyDMA example on page 45.

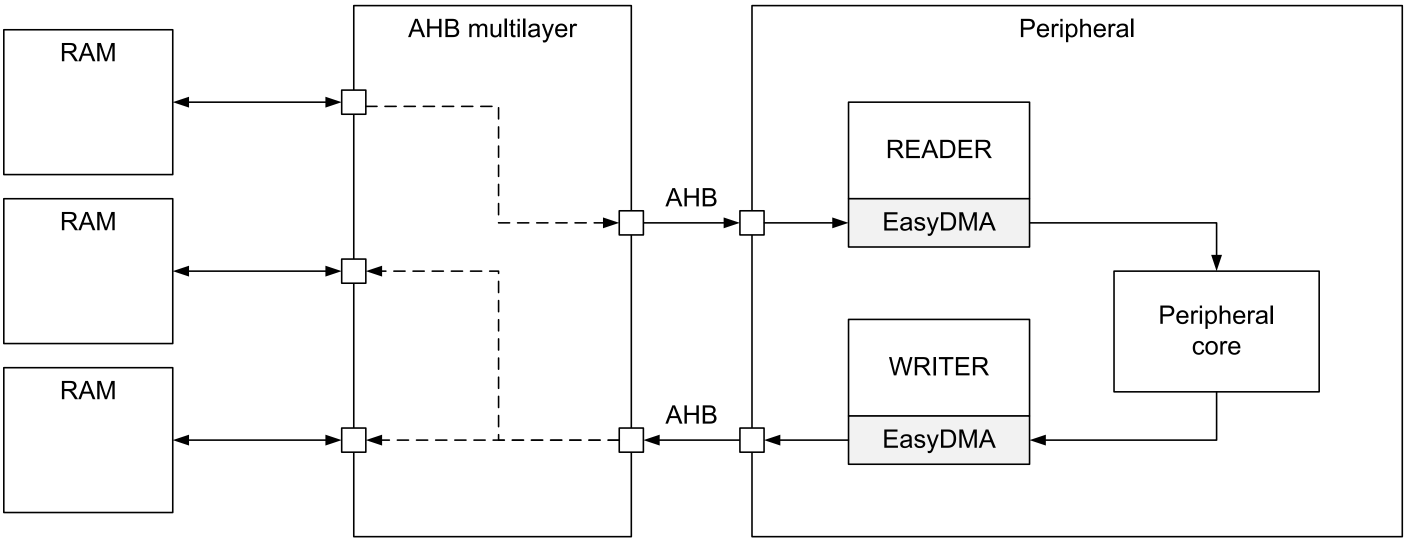

Figure 5: EasyDMA example

An EasyDMA channel is implemented in the following way, but some variations may occur:

```
READERBUFFER_SIZE 5
WRITERBUFFER_SIZE 6

uint8_t readerBuffer[READERBUFFER_SIZE]  __at__ 0x20000000;
uint8_t writerBuffer[WRITERBUFFER_SIZE]  __at__ 0x20000005;

// Configuring the READER channel
MYPERIPHERAL->READER.MAXCNT = READERBUFFER_SIZE;
MYPERIPHERAL->READER.PTR = &readerBuffer;

// Configure the WRITER channel
MYPERIPHERAL->WRITER.MAXCNT = WRITEERBUFFER_SIZE;
MYPERIPHERAL->WRITER.PTR = &writerBuffer;
```

This example shows a peripheral called MYPERIPHERAL that implements two EasyDMA channels - one for reading called READER, and one for writing called WRITER. When the peripheral is started, it is assumed that the peripheral will:

- Read 5 bytes from the readerBuffer located in RAM at address 0x20000000.
- Process the data.
- Write no more than 6 bytes back to the writerBuffer located in RAM at address 0x20000005.

The memory layout of these buffers is illustrated in EasyDMA memory layout on page 45.

0x20000000	readerBuffer[0]	readerBuffer[1]	readerBuffer[2]	readerBuffer[3]
0x20000004	readerBuffer[4]	writerBuffer[0]	writerBuffer[1]	writerBuffer[2]
0x20000008	writerBuffer[3]	writerBuffer[4]	writerBuffer[5]	

Figure 6: EasyDMA memory layout

The WRITER.MAXCNT register should not be specified larger than the actual size of the buffer (writerBuffer). Otherwise, the channel would overflow the writerBuffer.

Once an EasyDMA transfer is completed, the AMOUNT register can be read by the CPU to see how many bytes were transferred. For example, CPU can read MYPERIPHERAL->WRITER.AMOUNT register to see how many bytes WRITER wrote to RAM.

Note that the PTR register of a READER or WRITER must point to a valid memory region before use. The reset value of a PTR register is not guaranteed to point to valid memory. See Memory on page 20 for more information about the different memory regions and EasyDMA connectivity.

4.7.1 EasyDMA error handling

Some errors may occur during DMA handling.

If READER.PTR or WRITER.PTR is not pointing to a valid memory region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 20 for more information about the different memory regions.

If several AHB bus masters try to access the same AHB slave at the same time, AHB bus congestion might occur. An EasyDMA channel is an AHB master. Depending on the peripheral, the peripheral may either stall and wait for access to be granted, or lose data.

4.7.2 EasyDMA array list

EasyDMA is able to operate in Array List mode.

The Array List mode is implemented in channels where the LIST register is available.

The array list does not provide a mechanism to explicitly specify where the next item in the list is located. Instead, it assumes that the list is organized as a linear array where items are located one after the other in RAM.

The EasyDMA Array List can be implemented by using the data structure ArrayList_type as illustrated in the code example below using a READER EasyDMA channel as an example:

```
#define BUFFER_SIZE  4

typedef struct ArrayList
{
  uint8_t buffer[BUFFER_SIZE];
} ArrayList_type;

ArrayList_type ReaderList[3]  __at__ 0x20000000;

MYPERIPHERAL->READER.MAXCNT = BUFFER_SIZE;
MYPERIPHERAL->READER.PTR = &ReaderList;
MYPERIPHERAL->READER.LIST = MYPERIPHERAL_READER_LIST_ArrayList;
```

The data structure only includes a buffer with size equal to the size of READER.MAXCNT register. EasyDMA uses the READER.MAXCNT register to determine when the buffer is full.

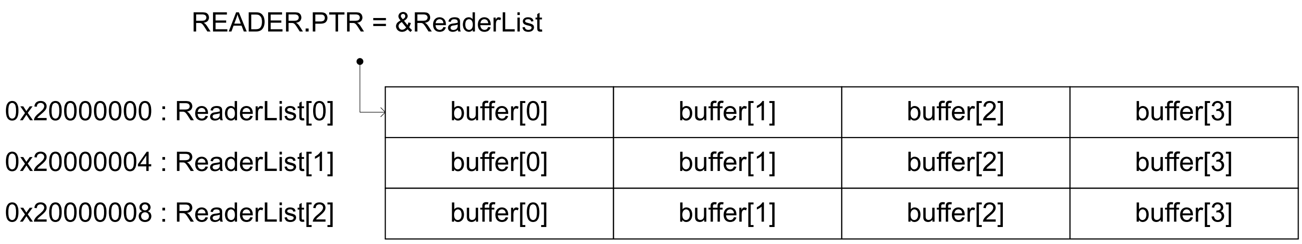

Figure 7: EasyDMA array list

4.8 AHB multilayer interconnect

On the AHB multilayer interconnect, the application CPU and all EasyDMA instances are AHB bus masters while RAM, cache and peripherals are AHB slaves. External MCU subsystems can be seen both as master and slave on the AHB multilayer interconnect.

Multiple AHB masters can access slave resources within the AHB multilayer interconnect as illustrated in Memory on page 20. Access rights to each of the AHB slaves are resolved using the natural priority of the different bus masters in the system.

5 Power and clock management

5.1 Functional description

The power and clock management system automatically ensures maximum power efficiency.

The core of the power and clock management is the power management unit (PMU) illustrated in the image below.

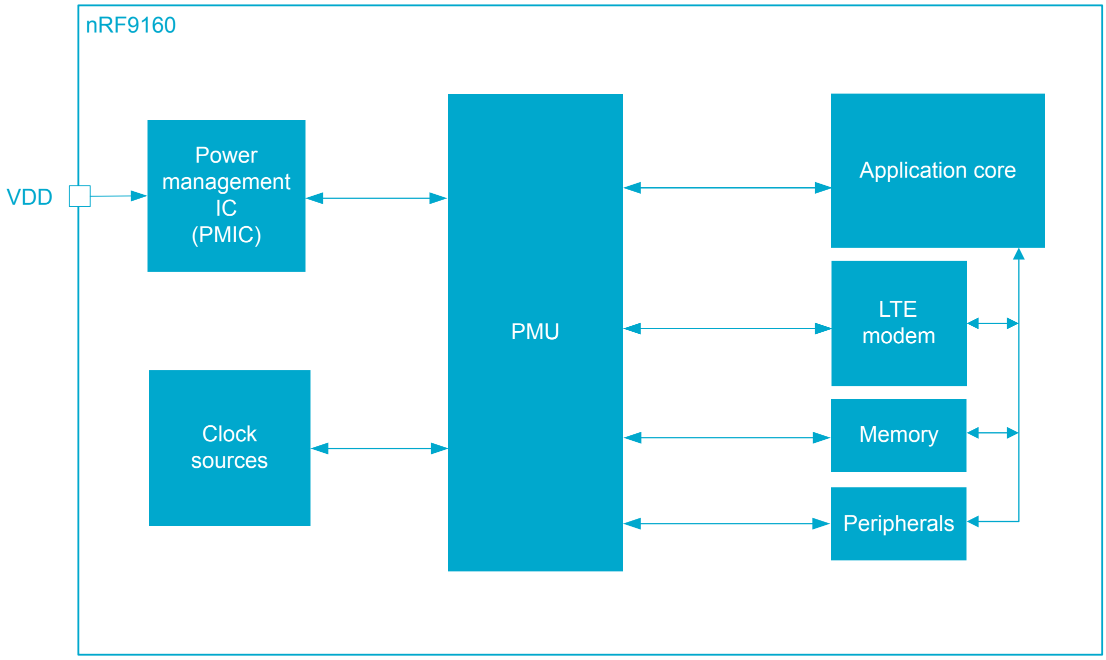

Figure 8: Power management unit

The PMU automatically tracks the power and clock resources required by the different components in the system. It then starts/stops and chooses operation modes in supply regulators and clock sources, without user interaction, to achieve the lowest power consumption possible.

5.1.1 Power management

The power management unit (PMU) handles two system-wide power modes - System ON and System OFF.

Internal blocks of the device are automatically powered by the PMU as they are required by the application.

5.1.1.1 System ON mode

System ON is the power mode entered after a power-on reset.

While in System ON, the system can reside in one of two sub modes:

- Low power
- Constant latency

The low power mode is default after power-on reset.

In low power mode, whenever no application or wireless activity takes place, function blocks like the application CPU, LTE modem and all peripherals are in IDLE state. That particular state is referred to as

System ON IDLE. In this state, all function blocks retain their state and configuration, so they are ready to become active once configured by the CPU.

If any application or modem activity occurs, the system leaves the System ON IDLE state. Once a given activity in a function block is completed, the system automatically returns to IDLE, retaining its configuration.

As long as the system resides in low power mode, the PMU ensures that the appropriate regulators and clock sources are started or stopped based on the needs of the function blocks active at any given time.

This automatic power management can be overridden by switching to constant latency mode. In this mode, the CPU wakeup latency and the PPI task response are constant and kept at a minimum. This is secured by keeping a set of base resources that are always enabled. The advantage of having a constant and predictable latency will be at the cost of having significantly increased power consumption compared to the low power mode. The constant latency mode is enabled by triggering the CONSTLAT task (TASKS_CONSTLAT on page 60).

While the system is in constant latency mode, the low power mode can be enabled by triggering LOWPWR task (TASKS_LOWPWR on page 60).

To reduce power consumption while in System ON IDLE, RAM blocks can be turned off in System ON mode while enabling the retention of these RAM blocks in RAM[n].POWER registers in VMC. RAM[n].POWER are retained registers, see Reset behavior on page 55. Note that these registers are usually overwritten by the startup code provided with the nRF application examples.

5.1.1.2 System OFF mode

System OFF is the deepest power saving mode the system can enter.

In this mode, the core system functionality is powered down and ongoing tasks terminated, and only the reset and the wakeup functions are available and responsive.

The device is put into System OFF mode using the REGULATORS register interface. When in System OFF mode, one of the following signals/actions will wake up the device:

1. DETECT signal, generated by the GPIO peripheral
2. RESET
3. start of debug session

When the device wakes up from System OFF mode, a system reset is performed.

One or more RAM blocks can be retained in System OFF mode depending on the settings in the RAM[n].POWER registers in VMC. RAM[n].POWER are retained registers, see Reset behavior on page 55. Note that these registers are usually overwritten by the startup code provided with the nRF application examples.

Before entering System OFF mode, the user must make sure that all on-going EasyDMA transactions have completed. This can be accomplished by making sure that EasyDMA enabled peripherals have stopped and END events from them received. The LTE modem also needs to be stopped, by issuing a command through the modem API, before entering System OFF mode. Once the command is issued, one should wait for the modem to respond that it actually has stopped, as there may be a delay until modem is disconnected from the network.

5.1.1.2.1 Emulated System OFF mode

If the device is in debug interface mode, System OFF will be emulated to secure that all required resources needed for debugging are available during System OFF.

See Overview on page 371 chapter for more information. Required resources needed for debugging include the following key components: Overview on page 371, CLOCK — Clock control on page 65, POWER — Power control on page 59, NVMC — Non-volatile memory controller on page 28, CPU on page 19, flash, and RAM. Since the CPU is kept on in emulated System OFF mode, it is required to add an infinite loop directly after entering System OFF, to prevent the CPU from executing code that normally should not be executed.

5.1.2 Power supply

The nRF9160 has a single main power supply VDD, and the internal components are powered by integrated voltage regulators. The PMU manages these regulators automatically, no voltage regulator control needs to be included in application firmware.

5.1.2.1 General purpose I/O supply

The input/output (I/O) drivers of P0.00 - P0.31 pins are supplied independently of VDD through VDD_GPIO. This enables easy match to signal voltage levels in the printed circuit board design.

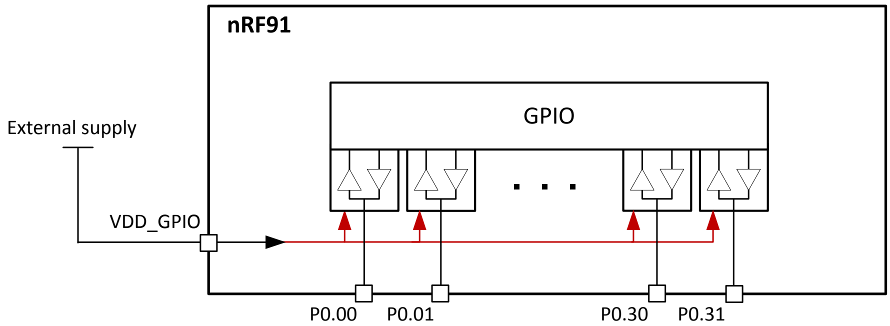

Figure 9: GPIO supply input (VDD_GPIO)

The I/Os are supplied via VDD_GPIO pin as shown in figure above. VDD_GPIO pin supports voltage levels within range given in table Operating conditions on page 391

5.1.3 Power supply monitoring

Power monitor solutions are available in the device, in order to survey the VDD (battery voltage).

5.1.3.1 Power supply supervisor

The power supply supervisor enables monitoring of the connected power supply.

Two functionalities are implemented:

- Power-on reset (POR): Generates a reset when the supply is applied to the device, and ensures that the device starts up in a known state
- Brownout reset (BOR): Generates a reset when the supply drops below the minimum voltage required for safe operations

Two BOR levels are used:

- V_{BOROFF}, used in System OFF
- V_{BORON}, used in System ON

The power supply supervisor is illustrated in the image below.

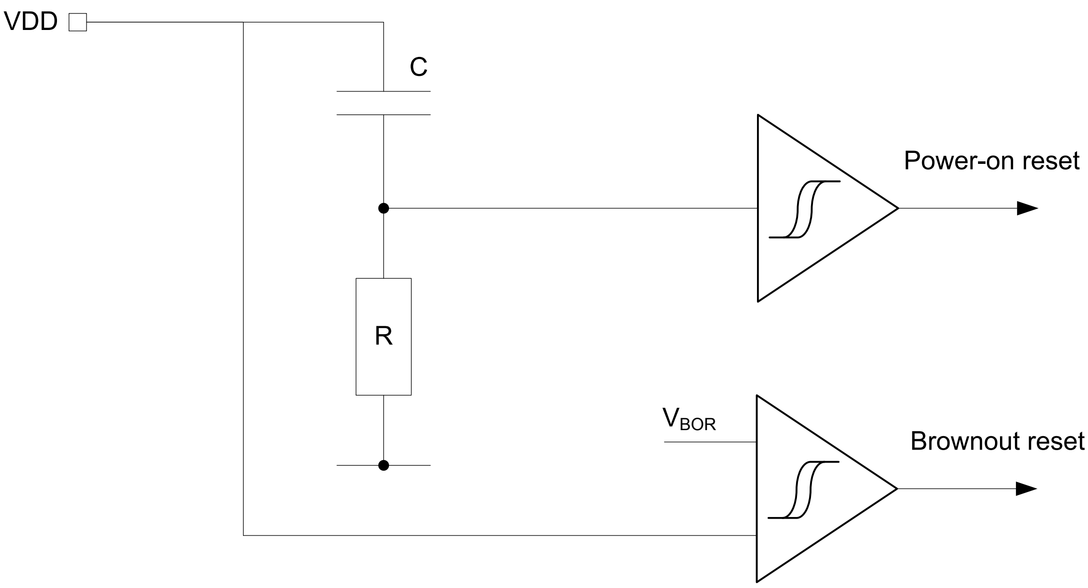

Figure 10: Power supply supervisor

5.1.3.2 Battery monitoring on VDD

A battery voltage (VDD) monitoring capability is provided via a modem API

> **Note:** For details regarding the modem API, please refer to *nRF Connect SDK* document and *nRF91 AT Commands, Command Reference Guide* document.

5.1.3.3 Electrical specification

5.1.3.3.1 Device startup times

Symbol	Description	Min.	Typ.	Max.	Units
t_{POR}	Time in power-on reset after VDD has reached V_{POR} (if ENABLE is tied to VDD) or after ENABLE input has been set to to logic high.	..	..	..	ms
t_{PINR}	The maximum time taken to pull up the nRESET pin and release reset after power-on reset. Dependent on the pin capacitive load (C)[2]: t=5RC, R=13 kΩ.	..	..	..	
$t_{PINR,500nF}$	C=500 nF	..	..	..	ms
$t_{PINR,10uF}$	C=10 µF	..	..	..	ms
t_{R2ON}	Time from reset to ON (CPU execute)				µs
t_{OFF2ON}	Time from OFF to CPU execute		36		µs
$t_{WFE2CPU}$	Time from WFE to CPU execute		22		µs
$t_{WFI2CPU}$	Time from WFI to CPU execute		33		µs
$t_{EVTSET,CL1}$	Time from HW event to PPI event in constant latency System ON mode	..	..	..	µs
$t_{EVTSET,CL0}$	Time from HW event to PPI event in low power System ON mode	..	..	..	µs

[2] To decrease the maximum time a device could be held in reset, a strong external pull-up resistor can be used.

5.1.3.3.2 Power supply supervisor

Symbol	Description	Min.	Typ.	Max.	Units
V_{BOR}	Brownout reset voltage threshold.		2.00		V
V_{POR}	Voltage threshold at which the device enters power-on reset (POR) when VDD is ramping up.			2.15	V

5.1.4 Clock management

The clock control system can source the system clocks from a range of high and low frequency oscillators, and distribute them to modules based upon a module's individual requirements. Clock generation and distribution is handled automatically by PMU to optimize current consumption.

Listed here are the available clock signal sources:

- 64 MHz oscillator (HFINT)
- 64 MHz high accuracy oscillator (HFXO)
- 32.768 kHz RC oscillator (LFRC)
- 32.768 kHz high accuracy oscillator (LFXO)

The clock and oscillator resources are configured and controlled via the CLOCK peripheral as illustrated below.

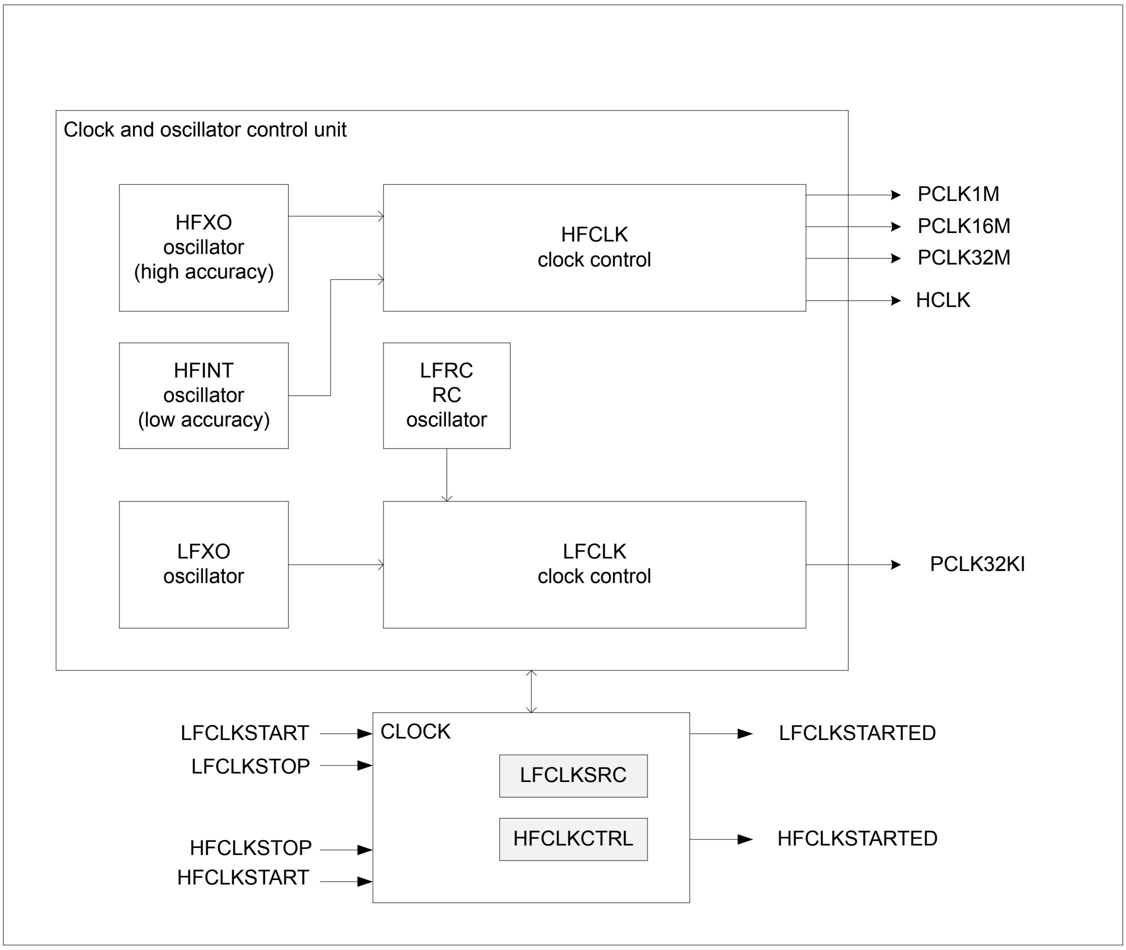

Figure 11: Clock and oscillator setup

5.1.4.1 HFCLK clock controller

The HFCLK clock controller provides several clocks in the system.

These are as follows:

- HCLK: 64 MHz CPU clock
- PCLK1M: 1 MHz peripheral clock
- PCLK16M: 16 MHz peripheral clock
- PCLK32M: 32 MHz peripheral clock

The HFCLK controller uses the following high frequency clock (HFCLK) sources:

- 64 MHz oscillator (HFINT)
- 64 MHz high accuracy oscillator (HFXO)

For illustration, see Clock and oscillator setup on page 52.

The HFCLK controller will automatically provide the clock(s) requested by the system. If the system does not request any clocks from the HFCLK controller, the controller will switch off all its clock sources and enter a power saving mode.

The HFINT source will be used when HFCLK is requested and HFXO has not been started.

The HFXO is started by triggering the HFCLKSTART task and stopped using the HFCLKSTOP task. A HFCLKSTARTED event will be generated when the HFXO has started and its frequency is stable.

5.1.4.2 LFCLK clock controller

The system supports several low frequency clock sources.

As illustrated in Clock and oscillator setup on page 52, the system supports the following low frequency clock sources:

- LFRC: 32.768 kHz RC oscillator
- LFXO: 32.768 kHz high accuracy oscillator

The LFCLK clock controller and all LFCLK clock sources are always switched off when in System OFF mode.

The LFCLK clock is started by first selecting the preferred clock source in the LFCLKSRC on page 72 register and then triggering the LFCLKSTART task. LFXO is highly recommended as the LFCLK clock source, since the LFRC has a large frequency variation.

> **Note:** The LTE modem requires using LFXO as the LFCLK source.

Switching between LFCLK clock sources can be done without stopping the LFCLK clock. A LFCLK clock source which is running prior to triggering the LFCLKSTART task will continue to run until the selected clock source has been available. After that the clock sources will be switched. Switching between clock sources will never introduce a glitch but it will stretch a clock pulse by 0.5 to 1.0 clock cycle (i.e. will delay rising edge by 0.5 to 1.0 clock cycle).

> **Note:** If the watchdog timer (WDT) is running, the default LFCLK clock source (LFRC - see LFCLKSRC on page 72) is started automatically (LFCLKSTART task doesn't have to be triggered).

A LFCLKSTARTED event will be generated when the selected LFCLK clock source has started.

> **Note:** When selecting LFXO as clock source for the first time, LFRC quality is provided until LFXO is stable.

A LFCLKSTOP task will stop global requesting of the LFCLK clock. However, if any system component (e.g. WDT, modem) requires the LFCLK, the clock won't be stopped. The LFCLKSTOP task should only be triggered after the STATE field in the LFCLKSTAT register indicates a LFCLK running-state.

5.1.4.2.1 32.768 kHz RC oscillator (LFRC)

The default source of the low frequency clock (LFCLK) is the 32.768 kHz RC oscillator (LFRC).

The LFRC frequency will be affected by variation in temperature.

5.1.4.3 Electrical specification

5.1.4.3.1 64 MHz internal oscillator (HFINT)

Symbol	Description	Min.	Typ.	Max.	Units
f_{NOM_HFINT}	Nominal output frequency		64		MHz
f_{TOL_HFINT}	Frequency tolerance		+-1	+-5	%
t_{START_HFINT}	Startup time		3.2		µs

5.1.4.3.2 64 MHz high accuracy oscillator (HFXO)

Symbol	Description	Min.	Typ.	Max.	Units
f_{NOM_HFXO}	Nominal output frequency		64		MHz
f_{TOL_HFXO}	Frequency tolerance		+-1		ppm
t_{START_HFXO}	Startup time		TBA		ms

5.1.4.3.3 32.768 kHz high accuracy oscillator (LFXO)

Symbol	Description	Min.	Typ.	Max.	Units
f_{NOM_LFXO}	Frequency		32.768		kHz
f_{TOL_LFXO}	Frequency tolerance		+-20		ppm
t_{START_LFXO}	Startup time		200		ms

5.1.4.3.4 32.768 kHz RC oscillator (LFRC)

Symbol	Description	Min.	Typ.	Max.	Units
f_{NOM_LFRC}	Nominal frequency		32.768		kHz
f_{TOL_LFRC}	Frequency tolerance		30		%
t_{START_LFRC}	Startup time		600		µs

5.1.5 Reset

There are multiple reset sources that may trigger a reset of the system. After a reset the CPU can query the RESETREAS (reset reason register) to find out which source generated the reset.

5.1.5.1 Power-on reset

The power-on reset generator initializes the system at power-on. The system is held in reset state until the supply has reached the minimum operating voltage and the internal voltage regulators have started.

5.1.5.2 Pin reset

A pin reset is generated when the physical reset pin (nRESET) on the device is pulled low.

To ensure that reset is issued correctly, the reset pin should be held low for time given in Pin reset on page 56 .

nRESET pin has an always-on internal pull-up resistor connected to nRF9160 internal voltage typically of 2.2 V level. The value of the pull-up resistor is given in Pin reset on page 56.

5.1.5.3 Wakeup from System OFF mode reset

The device is reset when it wakes up from System OFF mode.

The Debug access port is not reset following a wake up from System OFF mode if the device is in debug interface mode, see Overview on page 371 chapter for more information.

5.1.5.4 Soft reset

A soft reset is generated when the SYSRESETREQ bit of the application interrupt and reset control register (AIRCR register) in the ARM® core is set.

5.1.5.5 Watchdog reset

A watchdog reset is generated when the watchdog timer (WDT) times out.

See WDT — Watchdog timer on page 357 chapter for more information.

5.1.5.6 Brownout reset

The brownout reset generator puts the system in reset state if the supply voltage drops below the brownout reset threshold.

5.1.5.7 Retained registers

A retained register is a register that will retain its value in System OFF mode, and through a reset depending on reset source. See individual peripheral chapters for information of which registers are retained for the different peripherals.

5.1.5.8 Reset behavior

Reset behavior depends on the reset source.

The reset behavior is summarized in the table below.

Reset source	Reset target							
	CPU	Modem	Debug[3]	SWJ-DP	Not retained RAM[4]	Retained RAM[4]	WDT	RESETREAS
CPU lockup [5]	x	x						
Soft reset	x	x						
Wakeup from System OFF mode reset	x	x	x [6]		x		x	
Watchdog reset [7]	x	x	x		x		x	
Pin reset	x	x	x	x	x		x	
Brownout reset	x	x	x	x	x	x	x	x
Power-on reset	x	x	x	x	x	x	x	

Table 16: Reset behavior for the main components

3 All debug components excluding SWJ-DP. See Overview on page 371 chapter for more information about the different debug components in the system.
4 RAM can be configured to be retained using registers in VMC — Volatile memory controller on page 26
5 Reset from CPU lockup is disabled if the device is in debug interface mode. CPU lockup is not possible in System OFF.
6 The debug components will not be reset if the device is in debug interface mode.
7 Watchdog reset is not available in System OFF.

Note: The RAM is never reset but its content may be corrupted after reset in the cases given in the table above.

Reset source	Reset target					
	Regular peripheral registers	GPIO, SPU	NVMC WAITSTATENUM	NVMC IFCREADDELAY	REGULATORS, OSCILLATORS	POWER.GPREGRET
CPU lockup[5]	x	x	x			
Soft reset	x	x	x			
Wakeup from System OFF mode reset	x		x			
Watchdog reset[7]	x	x	x		x	
Pin reset	x	x	x		x	
Brownout reset	x	x	x	x	x	x
Power-on reset	x	x	x	x	x	x

Table 17: Reset behavior for the retained registers

5.1.5.9 Electrical specification

5.1.5.9.1 Pin reset

Symbol	Description	Min.	Typ.	Max.	Units
$t_{HOLDRESET}$	Hold time for reset pin when doing a pin reset	5			µs
$R_{PULL\text{-}UP}$	Value of the internal pull-up resistor				kΩ

5.2 Current consumption

As the system is being constantly tuned by the PMU described in Functional description on page 48, estimating the current consumption of an application can be challenging if the designer is not able to perform measurements directly on the hardware. To facilitate the estimation process, a set of current consumption scenarios are provided to show the typical current drawn from the VDD supply.

Each scenario specifies a set of operations and conditions applying to the given scenario. Current consumption scenarios, common conditions on page 56 shows a set of common conditions used in all scenarios, unless otherwise is stated in the description of a given scenario. Current consumption scenarios, common conditions on page 57 describes the conditions used for the modem current consumption specifications. All scenarios are listed in Electrical specification on page 57

Condition	Value
Supply	3.7 V
Temperature	25 °C
CPU	WFI (wait for interrupt)/WFE (wait for event) sleep
Peripherals	All idle
Clock	Not running
RAM	No retention
Cache enabled	Yes

Table 18: Current consumption scenarios, common conditions

Condition
Cat-M1 HD FDD mode
Ideal channel, no errors in DL/UL communication
Network response times at minimum
UICC current consumption excluded
Output power at antenna port, single-ended 50 Ω
All LTE modem current consumption numbers include application core idle mode consumption

Table 19: Current consumption scenarios, common conditions

5.2.1 Electrical specification

5.2.1.1 Sleep

Symbol	Description	Min.	Typ.	Max.	Units
$I_{MCUOFF0}$	MCU off, modem off, no RAM retention, wake on GPIO and reset		1.4		µA
I_{MCUON0}	MCU on IDLE, modem off, RTC off		1.8		µA
I_{MCUON1}	MCU on IDLE, modem off, RTC on		2.35		µA

5.2.1.2 Application CPU active current consumption

Symbol	Description	Min.	Typ.	Max.	Units
I_{CPU0_FLASH}	CPU running CoreMark @64 MHz from flash, clock = HFXO, cache enabled		2.88		mA
$I_{COREMARK\_PER\_MA\_FL}$	CoreMark per mA, executing from flash, CoreMark=243		84		CoreM mA
I_{CPU0_RAM}	CPU running CoreMark @64 MHz from RAM, clock = HFXO, cache enabled		2.32		mA
$I_{COREMARK\_PER\_MA\_RA}$	CoreMark per mA, executing from RAM, CoreMark=235		101		CoreM mA

5.2.1.3 I2S

Symbol	Description	Min.	Typ.	Max.	Units
I_{I2S0}	I2S transferring data @ 2 x 16 bit x 16 kHz (CONFIG.MCKFREQ = 32MDIV63, CONFIG.RATIO = 32X)		TBA		µA

5.2.1.4 PDM

Symbol	Description	Min.	Typ.	Max.	Units
I_{PDM}	PDM receiving and processing data @ 1 Msps		TBA		µA

5.2.1.5 PWM

Symbol	Description	Min.	Typ.	Max.	Units
I_{PWM0}	PWM running @ 125 kHz, fixed duty cycle		986.89		µA
I_{PWM1}	PWM running @ 16 MHz, fixed duty cycle		1160.77		µA

5.2.1.6 SAADC

Symbol	Description	Min.	Typ.	Max.	Units
I_{SAADC_HFXO}	SAADC sampling @ 16 ksps, with high accuracy clock HFXO, acquisition time = 20 µs		1300		µA
I_{SAADC_HFINT}	SAADC sampling @ 16 ksps, with low accuracy clock HFINT, acquisition time = 20 µs		302.34		µA

5.2.1.7 TIMER

Symbol	Description	Min.	Typ.	Max.	Units
I_{TIMER0}	TIMER running @ 1 MHz		772.87		µA
I_{TIMER1}	TIMER running @ 16 MHz		867.34		µA

5.2.1.8 SPIM

Symbol	Description	Min.	Typ.	Max.	Units
I_{SPIM0}	SPIM transferring data @ 2 Mbps, Clock = HFINT		1.2		mA
I_{SPIM1}	SPIM transferring data @ 2 Mbps, Clock = HFXO		1.9		mA
I_{SPIM2}	SPIM transferring data @ 8 Mbps, Clock = HFINT		1.5		mA
I_{SPIM3}	SPIM transferring data @ 8 Mbps, Clock = HFXO		2.2		mA

5.2.1.9 SPIS

Symbol	Description	Min.	Typ.	Max.	Units
I_{SPIS0}	SPIS transferring data @ 2 Mbps		TBA		µA

5.2.1.10 TWIM

Symbol	Description	Min.	Typ.	Max.	Units
I_{TWIM0}	TWIM running @ 100 kbps		TBA		µA

5.2.1.11 TWIS

Symbol	Description	Min.	Typ.	Max.	Units
$I_{TWIS,RUN0}$	TWIS transferring data @ 100 kbps		TBA		µA
$I_{TWIS1,RUN1}$	TWIS transferring data @ 400 kbps,		TBA		µA

5.2.1.12 UARTE

Symbol	Description	Min.	Typ.	Max.	Units
$I_{UARTE,1M}$	UARTE transferring data @ 1200 bps		1.5		mA
$I_{UARTE,115K}$	UARTE transferring data @ 115200 bps		1.3		mA

5.2.1.13 WDT

Symbol	Description	Min.	Typ.	Max.	Units
I_{WDT}	WDT started		3.95		µA

5.2.1.14 Modem current consumption

Symbol	Description	B13 (typ.)	B20 (typ.)	B3 (typ.)	B4 (typ.)	Units
Modem sleep current consumption						
I_{PSM}	PSM floor current	4	4	4	4	µA
Radio resource control (RRC) mode						
I_{EDRX}	eDRX average current, 81.92 s	19	19	19	19	µA
I_{IDRX}	Idle DRX average current, 2.56 s	199	199	199	199	µA
I_{RMC_0DBM}	Uplink 180 kbit/s, Pout 0 dBm, RMC settings as per 3GPP TS 36.521-1 Annex A.2	45	45	45	45	mA
I_{RMC_10DBM}	Uplink 180 kbit/s, Pout 10 dBm, RMC settings as per 3GPP TS 36.521-1 Annex A.2	50	50	55	55	mA
I_{RMC_23DBM}	Uplink 180 kbit/s, Pout 23 dBm, RMC settings as per 3GPP TS 36.521-1 Annex A.2	105	110	140	140	mA
Modem active current consumption						
I_{TX_0DBM}	TX subframe, Pout 0 dBm	60	60	65	65	mA
I_{TX_10DBM}	TX subframe, Pout 10 dBm	80	85	95	90	mA
I_{TX_23DBM}	TX subframe, Pout 23 dBm	255	275	380	365	mA
I_{RX_-90DBM}	RX subframe, Pin -90 dBm	45	45	45	45	mA
$I_{TX\_TRANSIENT}$	TX transient	40	45	50	50	mA/µs
Modem peak current consumption						
I_{TX_PEAK}	TX subframe, Pout >21 dBm, Ant VSWR3	335	360	455	450	mA
I_{TX_PEAK}	TX subframe, Pout >20 dBm, Ant VSWR3, Vbat 3.5 V, Temp 85 °C	350	380	460	450	mA
I_{TX_PEAK}	TX subframe, Pout >20 dBm, Ant VSWR3, Vbat 3.0 V, Temp 85 °C	410	445	535	525	mA

5.2.1.15 GPS current consumption

Symbol	Description	Min.	Typ.	Max.	Units
$I_{GPS\_CONTINUOUS}$	Continuous tracking, typical peak current without power saving mode		47		mA
$I_{GPS\_CONTINUOUS\_PSM}$	Continuous tracking, power saving mode		12		mA
I_{GPS_SINGLE}	Single shot, one fix every 2 minutes		1.3		mA

5.3 Register description

5.3.1 POWER — Power control

The POWER module provides an interface to tasks, events, interrupt and reset related configuration settings of the power management unit.

Note: Registers INTEN on page 63, INTENSET on page 63, and INTENCLR on page 64 are the same registers (at the same address) as corresponding registers in CLOCK — Clock control on page 65.

5.3.1.1 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50005000 0x40005000	POWER	POWER : S POWER : NS	US	NA	Power control	

Table 20: Instances

Register	Offset	Security	Description
TASKS_CONSTLAT	0x78		Enable constant latency mode.
TASKS_LOWPWR	0x7C		Enable low power mode (variable latency)
SUBSCRIBE_CONSTLAT	0xF8		Subscribe configuration for task CONSTLAT
SUBSCRIBE_LOWPWR	0xFC		Subscribe configuration for task LOWPWR
EVENTS_POFWARN	0x108		Power failure warning
EVENTS_SLEEPENTER	0x114		CPU entered WFI/WFE sleep
EVENTS_SLEEPEXIT	0x118		CPU exited WFI/WFE sleep
PUBLISH_POFWARN	0x188		Publish configuration for event POFWARN
PUBLISH_SLEEPENTER	0x194		Publish configuration for event SLEEPENTER
PUBLISH_SLEEPEXIT	0x198		Publish configuration for event SLEEPEXIT
INTEN	0x300		Enable or disable interrupt
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
RESETREAS	0x400		Reset reason
POWERSTATUS	0x440		Modem domain power status
GPREGRET[0]	0x51C		General purpose retention register
GPREGRET[1]	0x520		General purpose retention register

Table 21: Register overview

5.3.1.1.1 TASKS_CONSTLAT

Address offset: 0x78

Enable constant latency mode.

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_CONSTLAT			Enable constant latency mode.
			Trigger	1	Trigger task

5.3.1.1.2 TASKS_LOWPWR

Address offset: 0x7C

Enable low power mode (variable latency)

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_LOWPWR			Enable low power mode (variable latency)
			Trigger	1	Trigger task

5.3.1.1.3 SUBSCRIBE_CONSTLAT

Address offset: 0xF8

Subscribe configuration for task CONSTLAT

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task CONSTLAT will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

5.3.1.1.4 SUBSCRIBE_LOWPWR

Address offset: 0xFC

Subscribe configuration for task LOWPWR

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task LOWPWR will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

5.3.1.1.5 EVENTS_POFWARN

Address offset: 0x108

Power failure warning

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_POFWARN			Power failure warning
			NotGenerated	0	Event not generated
			Generated	1	Event generated

5.3.1.1.6 EVENTS_SLEEPENTER

Address offset: 0x114

CPU entered WFI/WFE sleep

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_SLEEPENTER			CPU entered WFI/WFE sleep
			NotGenerated	0	Event not generated
			Generated	1	Event generated

5.3.1.1.7 EVENTS_SLEEPEXIT

Address offset: 0x118

CPU exited WFI/WFE sleep

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_SLEEPEXIT			CPU exited WFI/WFE sleep
			NotGenerated	0	Event not generated
			Generated	1	Event generated

5.3.1.1.8 PUBLISH_POFWARN

Address offset: 0x188

Publish configuration for event POFWARN

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event POFWARN will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

5.3.1.1.9 PUBLISH_SLEEPENTER

Address offset: 0x194

Publish configuration for event SLEEPENTER

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event SLEEPENTER will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

5.3.1.1.10 PUBLISH_SLEEPEXIT

Address offset: 0x198

Publish configuration for event SLEEPEXIT

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event SLEEPEXIT will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

5.3.1.1.11 INTEN

Address offset: 0x300

Enable or disable interrupt

Bit number	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
ID																										D	C			A		
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	POFWARN			Enable or disable interrupt for event POFWARN
			Disabled	0	Disable
			Enabled	1	Enable
C	RW	SLEEPENTER			Enable or disable interrupt for event SLEEPENTER
			Disabled	0	Disable
			Enabled	1	Enable
D	RW	SLEEPEXIT			Enable or disable interrupt for event SLEEPEXIT
			Disabled	0	Disable
			Enabled	1	Enable

5.3.1.1.12 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID																										D	C			A		
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	POFWARN			Write '1' to enable interrupt for event POFWARN
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	SLEEPENTER			Write '1' to enable interrupt for event SLEEPENTER
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
D	RW	SLEEPEXIT			Write '1' to enable interrupt for event SLEEPEXIT
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

5.3.1.1.13 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID																										D	C			A		
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	POFWARN			Write '1' to disable interrupt for event POFWARN
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	SLEEPENTER			Write '1' to disable interrupt for event SLEEPENTER
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
D	RW	SLEEPEXIT			Write '1' to disable interrupt for event SLEEPEXIT
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

5.3.1.1.14 RESETREAS

Address offset: 0x400

Reset reason

Unless cleared, the RESETREAS register will be cumulative. A field is cleared by writing '1' to it. If none of the reset sources are flagged, this indicates that the chip was reset from the on-chip reset generator, which will indicate a power-on reset or a brownout reset.

Bit number	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
ID														G	F	E												D		C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	RESETPIN			Reset from pin reset detected
			NotDetected	0	Not detected
			Detected	1	Detected
B	RW	DOG			Reset from global watchdog detected
			NotDetected	0	Not detected
			Detected	1	Detected
C	RW	OFF			Reset due to wakeup from System OFF mode, when wakeup is triggered by DETECT signal from GPIO
			NotDetected	0	Not detected
			Detected	1	Detected
D	RW	DIF			Reset due to wakeup from System OFF mode, when wakeup is triggered by entering debug interface mode
			NotDetected	0	Not detected
			Detected	1	Detected
E	RW	SREQ			Reset from AIRCR.SYSRESETREQ detected
			NotDetected	0	Not detected
			Detected	1	Detected
F	RW	LOCKUP			Reset from CPU lock-up detected

Bit number	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
ID														G	F	E												D		C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			NotDetected	0	Not detected
			Detected	1	Detected
G	RW	CTRLAP			Reset triggered through CTRL-AP
			NotDetected	0	Not detected
			Detected	1	Detected

5.3.1.1.15 POWERSTATUS

Address offset: 0x440

Modem domain power status

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	LTEMODEM			LTE modem domain status
			OFF	0	LTE modem domain is powered off
			ON	1	LTE modem domain is powered on

5.3.1.1.16 GPREGRET[n] (n=0..1)

Address offset: 0x51C + (n × 0x4)

General purpose retention register

Bit number	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
ID																									A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	GPREGRET			General purpose retention register This register is a retained register

5.3.2 CLOCK — Clock control

The CLOCK module provides one of the interfaces to power and clock management configuration settings.

Through CLOCK module it is able to configure the following:

- LFCLK clock source setup
- LFCLK and HFCLK status
- Tasks and events
- Interrupts
- Reset

> **Note:** Registers INTEN on page 69, INTENSET on page 70, and INTENCLR on page 70 are the same registers (at the same address) as corresponding registers in POWER — Power control on page 59.

5.3.2.1 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50005000 0x40005000	CLOCK	CLOCK : S CLOCK : NS	US	NA	Clock control	

Table 22: Instances

Register	Offset	Security	Description
TASKS_HFCLKSTART	0x000		Start HFCLK source
TASKS_HFCLKSTOP	0x004		Stop HFCLK source
TASKS_LFCLKSTART	0x008		Start LFCLK source
TASKS_LFCLKSTOP	0x00C		Stop LFCLK source
SUBSCRIBE_HFCLKSTART	0x080		Subscribe configuration for task HFCLKSTART
SUBSCRIBE_HFCLKSTOP	0x084		Subscribe configuration for task HFCLKSTOP
SUBSCRIBE_LFCLKSTART	0x088		Subscribe configuration for task LFCLKSTART
SUBSCRIBE_LFCLKSTOP	0x08C		Subscribe configuration for task LFCLKSTOP
EVENTS_HFCLKSTARTED	0x100		HFCLK oscillator started
EVENTS_LFCLKSTARTED	0x104		LFCLK started
PUBLISH_HFCLKSTARTED	0x180		Publish configuration for event HFCLKSTARTED
PUBLISH_LFCLKSTARTED	0x184		Publish configuration for event LFCLKSTARTED
INTEN	0x300		Enable or disable interrupt
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
INTPEND	0x30C		Pending interrupts
HFCLKRUN	0x408		Status indicating that HFCLKSTART task has been triggered
HFCLKSTAT	0x40C		The register shows if HFXO has been requested by triggering HFCLKSTART task and if it has been started (STATE)
LFCLKRUN	0x414		Status indicating that LFCLKSTART task has been triggered
LFCLKSTAT	0x418		The register shows which LFCLK source has been requested (SRC) when triggering LFCLKSTART task and if the source has been started (STATE)
LFCLKSRCCOPY	0x41C		Copy of LFCLKSRC register, set after LFCLKSTART task has been triggered
LFCLKSRC	0x518		Clock source for the LFCLK. LFCLKSTART task starts starts a clock source selected with this register.

Table 23: Register overview

5.3.2.1.1 TASKS_HFCLKSTART

Address offset: 0x000

Start HFCLK source

Bit number	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
ID	A (bit 0)
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_HFCLKSTART			Start HFCLK source
			Trigger	1	Trigger task

5.3.2.1.2 TASKS_HFCLKSTOP

Address offset: 0x004

Stop HFCLK source

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_HFCLKSTOP			Stop HFCLK source
			Trigger	1	Trigger task

5.3.2.1.3 TASKS_LFCLKSTART

Address offset: 0x008

Start LFCLK source

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_LFCLKSTART			Start LFCLK source
			Trigger	1	Trigger task

5.3.2.1.4 TASKS_LFCLKSTOP

Address offset: 0x00C

Stop LFCLK source

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_LFCLKSTOP			Stop LFCLK source
			Trigger	1	Trigger task

5.3.2.1.5 SUBSCRIBE_HFCLKSTART

Address offset: 0x080

Subscribe configuration for task HFCLKSTART

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task HFCLKSTART will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

5.3.2.1.6 SUBSCRIBE_HFCLKSTOP

Address offset: 0x084

Subscribe configuration for task HFCLKSTOP

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task HFCLKSTOP will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

5.3.2.1.7 SUBSCRIBE_LFCLKSTART

Address offset: 0x088

Subscribe configuration for task LFCLKSTART

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task LFCLKSTART will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

5.3.2.1.8 SUBSCRIBE_LFCLKSTOP

Address offset: 0x08C

Subscribe configuration for task LFCLKSTOP

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task LFCLKSTOP will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

5.3.2.1.9 EVENTS_HFCLKSTARTED

Address offset: 0x100

HFCLK oscillator started

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_HFCLKSTARTED			HFCLK oscillator started
			NotGenerated	0	Event not generated
			Generated	1	Event generated

5.3.2.1.10 EVENTS_LFCLKSTARTED

Address offset: 0x104

LFCLK started

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_LFCLKSTARTED			LFCLK started
			NotGenerated	0	Event not generated
			Generated	1	Event generated

5.3.2.1.11 PUBLISH_HFCLKSTARTED

Address offset: 0x180

Publish configuration for event HFCLKSTARTED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event HFCLKSTARTED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

5.3.2.1.12 PUBLISH_LFCLKSTARTED

Address offset: 0x184

Publish configuration for event LFCLKSTARTED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event LFCLKSTARTED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

5.3.2.1.13 INTEN

Address offset: 0x300

Enable or disable interrupt

Bit number					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
ID					B A
Reset 0x00000000					**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
ID	Acce	Field	Value ID	Value	Description
A	RW	HFCLKSTARTED			Enable or disable interrupt for event HFCLKSTARTED
			Disabled	0	Disable
			Enabled	1	Enable
B	RW	LFCLKSTARTED			Enable or disable interrupt for event LFCLKSTARTED
			Disabled	0	Disable
			Enabled	1	Enable

5.3.2.1.14 INTENSET

Address offset: 0x304

Enable interrupt

Bit number					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
ID					B A
Reset 0x00000000					**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
ID	Acce	Field	Value ID	Value	Description
A	RW	HFCLKSTARTED			Write '1' to enable interrupt for event HFCLKSTARTED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	LFCLKSTARTED			Write '1' to enable interrupt for event LFCLKSTARTED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

5.3.2.1.15 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number					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
ID					B A
Reset 0x00000000					**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
ID	Acce	Field	Value ID	Value	Description
A	RW	HFCLKSTARTED			Write '1' to disable interrupt for event HFCLKSTARTED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	LFCLKSTARTED			Write '1' to disable interrupt for event LFCLKSTARTED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

5.3.2.1.16 INTPEND

Address offset: 0x30C

Pending interrupts

Bit number	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
ID																															B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	HFCLKSTARTED			Read pending status of interrupt for event HFCLKSTARTED
			NotPending	0	Read: Not pending
			Pending	1	Read: Pending
B	R	LFCLKSTARTED			Read pending status of interrupt for event LFCLKSTARTED
			NotPending	0	Read: Not pending
			Pending	1	Read: Pending

5.3.2.1.17 HFCLKRUN

Address offset: 0x408

Status indicating that HFCLKSTART task has been triggered

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	STATUS			HFCLKSTART task triggered or not
			NotTriggered	0	Task not triggered
			Triggered	1	Task triggered

5.3.2.1.18 HFCLKSTAT

Address offset: 0x40C

The register shows if HFXO has been requested by triggering HFCLKSTART task and if it has been started (STATE)

Bit number	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
ID																B																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	SRC			Active clock source
			HFXO	1	HFXO - 64 MHz clock derived from external 32 MHz crystal oscillator
B	R	STATE			HFCLK state
			NotRunning	0	HFXO has not been started or HFCLKSTOP task has been triggered
			Running	1	HFXO has been started (HFCLKSTARTED event has been generated)

5.3.2.1.19 LFCLKRUN

Address offset: 0x414

Status indicating that LFCLKSTART task has been triggered

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	STATUS			LFCLKSTART task triggered or not
			NotTriggered	0	Task not triggered
			Triggered	1	Task triggered

5.3.2.1.20 LFCLKSTAT

Address offset: 0x418

The register shows which LFCLK source has been requested (SRC) when triggering LFCLKSTART task and if the source has been started (STATE)

Bit number	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
ID																B															A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	SRC			Active clock source
			RFU	0	Reserved for future use
			LFRC	1	32.768 kHz RC oscillator
			LFXO	2	32.768 kHz crystal oscillator
B	R	STATE			LFCLK state
			NotRunning	0	Requested LFCLK source has not been started or LFCLKSTOP task has been triggered
			Running	1	Requested LFCLK source has been started (LFCLKSTARTED event has been generated)

5.3.2.1.21 LFCLKSRCCOPY

Address offset: 0x41C

Copy of LFCLKSRC register, set after LFCLKSTART task has been triggered

Bit number	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
ID																															A	A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	SRC			Clock source
			RFU	0	Reserved for future use
			LFRC	1	32.768 kHz RC oscillator
			LFXO	2	32.768 kHz crystal oscillator

5.3.2.1.22 LFCLKSRC

Address offset: 0x518

Clock source for the LFCLK. LFCLKSTART task starts starts a clock source selected with this register.

Bit number	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
ID																															A	A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	SRC			Clock source
			RFU	0	Reserved for future use (equals selecting LFRC)
			LFRC	1	32.768 kHz RC oscillator
			LFXO	2	32.768 kHz crystal oscillator

5.3.3 REGULATORS — Voltage regulators control

The REGULATORS module provides an interface to certain configuration settings of on-chip voltage regulators.

5.3.3.1 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50004000 0x40004000	REGULATORS	REGULATORS : S REGULATORS : NS	US	NA	Regulator configuration	

Table 24: Instances

Register	Offset	Security	Description
SYSTEMOFF	0x500		System OFF register
DCDCEN	0x578		Enable DC/DC mode of the main voltage regulator.

Table 25: Register overview

5.3.3.1.1 SYSTEMOFF

Address offset: 0x500

System OFF register

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	SYSTEMOFF			Enable System OFF mode
			Enable	1	Enable System OFF mode

5.3.3.1.2 DCDCEN

Address offset: 0x578

Enable DC/DC mode of the main voltage regulator.

Note: DCDCEN must be set to 1 (enabled) before the LTE modem is started.

Bit number			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
ID																																		A
Reset 0x00000000			**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	DCDCEN			Enable DC/DC converter
			Disabled	0	DC/DC mode is disabled
			Enabled	1	DC/DC mode is enabled

6 Peripherals

6.1 CRYPTOCELL — ARM TrustZone CryptoCell 310

ARM® TrustZone® CryptoCell 310 (CRYPTOCELL) is a security subsystem which provides root of trust (RoT) and cryptographic services for a device.

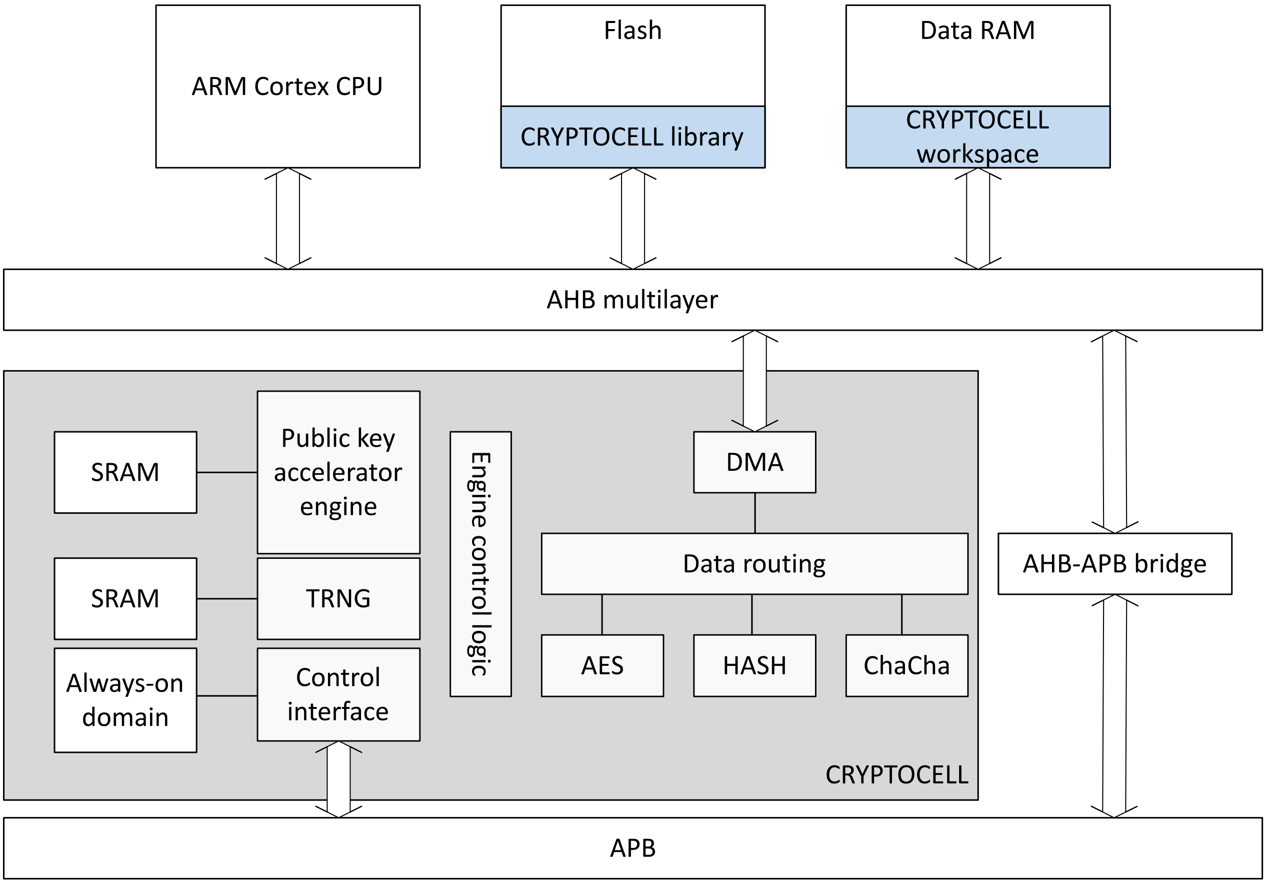

Figure 12: Block diagram for CRYPTOCELL

The following cryptographic features are provided:

- True random number generator (TRNG) compliant with NIST 800-90B, AIS-31, and FIPS 140-2
- Pseudorandom number generator (PRNG) using underlying AES engine compliant with NIST 800-90A
- RSA public key cryptography
  - Up to 2048-bit key size
  - PKCS#1 v2.1/v1.5
  - Optional CRT support
- Elliptic curve cryptography (ECC)
  - NIST FIPS 186-4 recommended curves using pseudorandom parameters, up to 521 bits:
    - Prime field: P-192, P-224, P-256, P-384, P-521
  - SEC 2 recommended curves using pseudorandom parameters, up to 521 bits:
    - Prime field: secp160r1, secp192r1, secp224r1, secp256r1, secp384r1, secp521r1
  - Koblitz curves using fixed parameters, up to 256 bits:

- Prime field: secp160k1, secp192k1, secp224k1, secp256k1
- Edwards/Montgomery curves:
  - Ed25519, Curve25519
- ECDH/ECDSA support
- Secure remote password protocol (SRP)
  - Up to 3072-bit operations
- Hashing functions
  - SHA-1, SHA-2 up to 256 bits
  - Keyed-hash message authentication code (HMAC)
- AES symmetric encryption
  - General purpose AES engine (encrypt/decrypt, sign/verify)
  - 128-bit key size
  - Supported encryption modes: ECB, CBC, CMAC/CBC-MAC, CTR, CCM/CCM* (CCM* is a minor variation of CCM)
- ChaCha20/Poly1305 symmetric encryption
  - Supported key size: 128 and 256 bits
  - Authenticated encryption with associated data (AEAD) mode

6.1.1 Usage

The CRYPTOCELL state is controlled via a register interface. The cryptographic functions of CRYPTOCELL are accessible by using a software library provided in the device SDK, not directly via a register interface.

To enable CRYPTOCELL, use register ENABLE on page 78.

> **Note:** Keeping the CRYPTOCELL subsystem enabled will prevent the device from reaching the System ON, All Idle state.

6.1.2 Always-on (AO) power domain

The CRYPTOCELL subsystem has an internal always-on (AO) power domain for retaining device secrets when CRYPTOCELL is disabled.

The following information is retained by the AO power domain:

- 4 bits indicating the configured CRYPTOCELL lifecycle state (LCS)
- 1 bit indicating if the hard-coded RTL key, K_{PRTL} (see RTL key on page 77), is available for use
- 128-bit device root key, K_{DR} (see Device root key on page 77)

A reset from any reset source will erase the content in the AO power domain.

6.1.3 Lifecycle state (LCS)

Lifecycle refers to multiple states a device goes through during its lifetime. Two valid lifecycle states are offered for the device - debug and secure.

The CRYPTOCELL subsystem lifecycle state (LCS) is controlled through register HOST_IOT_LCS on page 81. A valid LCS is configured by writing either value `Debug` or `Secure` into the LCS field of this register. A correctly configured LCS can be validated by reading back the read-only field LCS_IS_VALID from the abovementioned register. The LCS_IS_VALID field value will change from `Invalid` to `Valid` once a valid LCS value has been written.

LCS field value	LCS_IS_VALID field value	Description
Secure	Invalid	Default reset value indicating that LCS has not been configured.
Secure	Valid	LCS set to secure mode, and LCS is valid. Registers HOST_IOT_KDR[0..3] can only be written once per reset cycle. Any additional writes will be ignored.
Debug	Valid	LCS set to debug mode, and LCS is valid. Registers HOST_IOT_KDR[0..3] can be written multiple times.

Table 26: Lifecycle states

6.1.4 Cryptographic key selection

The CRYPTOCELL subsystem can be instructed to operate on different cryptographic keys.

Through register HOST_CRYPTOKEY_SEL on page 79, the following key types can be selected for cryptographic operations:

- RTL key K_{PRTL}
- Device root key K_{DR}
- Session key

K_{PRTL} and K_{DR} are configured as part of the CRYPTOCELL initialization process, while session keys are provided by the application through the software library API.

6.1.4.1 RTL key

The ARM® TrustZone® CryptoCell 310 contains one hard-coded RTL key referred to as K_{PRTL}. This key is set to the same value for all devices with the same part code in the hardware design and cannot be changed.

The K_{PRTL} key can be requested for use in cryptographic operations by the CRYPTOCELL, without revealing the key value itself. Access to use of K_{PRTL} in cryptographic operations can be disabled until next reset by writing to register HOST_IOT_KPRTL_LOCK on page 80. If a locked K_{PRTL} key is requested for use, a zero vector key will be routed to the AES engine instead.

6.1.4.2 Device root key

The device root key K_{DR} is a 128-bit AES key programmed into the CRYPTOCELL subsystem using firmware. It is retained in the AO power domain until the next reset.

Once configured, it is possible to perform cryptographic operations using the the CRYPTOCELL subsystem where K_{DR} is selected as key input without having access to the key value itself. The K_{DR} key value must be written to registers HOST_IOT_KDR[0..3]. These 4 registers are write-only if LCS is set to debug mode, and write-once if LCS is set to secure mode. The K_{DR} key value is successfully retained when the read-back value of register HOST_IOT_KDR0 on page 80 changes to 1.

6.1.5 Direct memory access (DMA)

The CRYPTOCELL subsystem implements direct memory access (DMA) for accessing memory without CPU intervention.

Any data stored in memory type(s) not accessible by the DMA engine must be copied to SRAM before it can be processed by the CRYPTOCELL subsystem. Maximum DMA transaction size is limited to 2^{16}-1 bytes.

6.1.6 Standards

ARM® TrustZone® CryptoCell 310 (CRYPTOCELL) supports a number of cryptography standards.

Algorithm family	Identification code	Document title
TRNG	NIST SP 800-90B	*Recommendation for the Entropy Sources Used for Random Bit Generation*
	AIS-31	*A proposal for: Functionality classes and evaluation methodology for physical random number generators*
	FIPS 140-2	*Security Requirements for Cryptographic Modules*
PRNG	NIST SP 800-90A	*Recommendation for Random Number Generation Using Deterministic Random Bit Generators*
Stream cipher	Chacha	*ChaCha, a variant of Salsa20*, Daniel J. Bernstein, January 28th 2008
MAC	Poly1305	*The Poly1305-AES message-authentication code*, Daniel J. Bernstein *Cryptography in NaCl*, Daniel J. Bernstein
Key agreement	SRP	*The Secure Remote Password Protocol*, Thomas Wu, November 11th 1997
AES	FIPS-197	*Advanced Encryption Standard (AES)*
	NIST SP 800-38A	*Recommendation for Block Cipher Modes of Operation - Methods and Techniques*
	NIST SP 800-38B	*Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*
	NIST SP 800-38C	*Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality*
	ISO/IEC 9797-1	AES CBC-MAC per ISO/IEC 9797-1 MAC algorithm 1
	IEEE 802.15.4-2011	*IEEE Standard for Local and metropolitan area networks - Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)*, Annex B.4: *Specification of generic CCM* mode of operation*
Hash	FIPS 180-3	Secure Hash Standard (SHA1, SHA-224, SHA-256)
	RFC2104	*HMAC: Keyed-Hashing for Message Authentication*
RSA	PKCS#1	*Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications* v1.5/2.1
Diffie-Hellman	ANSI X9.42	*Public Key Cryptography for the Financial Services Industry: Agreement of Symmetric Keys Using Discrete Logarithm Cryptography*
	PKCS#3	*Diffie-Hellman Key-Agreement Standard*
ECC	ANSI X9.63	*Public Key Cryptography for the Financial Services Industry - Key Agreement and Key Transport Using Elliptic Curve Cryptography*
	IEEE 1363	*Standard Specifications for Public-Key Cryptography*
	ANSI X9.62	*Public Key Cryptography For The Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*
	Ed25519	Edwards-curve, *Ed25519: high-speed high-security signatures*, Daniel J. Bernstein, Niels Duif, Tanja Lange, Peter Schwabe, and Bo-Yin Yang
	Curve25519	Montgomery curve, *Curve25519: new Diffie-Hellman speed records*, Daniel J. Bernstein
	FIPS 186-4	*Digital Signature Standard (DSS)*
	SEC 2	*Recommended Elliptic Curve Domain Parameters*, Certicom Research
	NIST SP 800-56A rev. 2	*Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography*

Table 27: CRYPTOCELL cryptography standards

6.1.7 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50840000	CRYPTOCELL	CRYPTOCELL	S	NSA	CryptoCell sub-system control interface	

Table 28: Instances

Register	Offset	Security	Description
ENABLE	0x500		Enable CRYPTOCELL subsystem

Table 29: Register overview

6.1.7.1 ENABLE

Address offset: 0x500

Enable CRYPTOCELL subsystem

Bit number				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
ID				A
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
ID	Acce Field	Value ID	Value	Description
A	RW ENABLE			Enable or disable the CRYPTOCELL subsystem
		Disabled	0	CRYPTOCELL subsystem disabled
		Enabled	1	CRYPTOCELL subsystem enabled. When enabled the CRYPTOCELL subsystem can be initialized and controlled through the CryptoCell firmware API.

6.1.8 Host interface

This chapter describes host registers used to control the CRYPTOCELL subsystem behavior.

6.1.8.1 HOST_RGF block

The HOST_RGF block contains registers for configuring LCS and device root key K_{DR}, in addition to selecting which cryptographic key is connected to the AES engine.

6.1.8.1.1 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50840000	CC_HOST_RGF	CC_HOST_RGF	S	NSA	Host platform interface	

Table 30: Instances

Register	Offset	Security	Description
HOST_CRYPTOKEY_SEL	0x1A38		AES hardware key select
HOST_IOT_KPRTL_LOCK	0x1A4C		This write-once register is the K_PRTL lock register. When this register is set, K_PRTL cannot be used and a zeroed key will be used instead. The value of this register is saved in the CRYPTOCELL AO power domain.
HOST_IOT_KDR0	0x1A50		This register holds bits 31:0 of K_DR. The value of this register is saved in the CRYPTOCELL AO power domain. Reading from this address returns the K_DR valid status indicating if K_DR is successfully retained.
HOST_IOT_KDR1	0x1A54		This register holds bits 63:32 of K_DR. The value of this register is saved in the CRYPTOCELL AO power domain.
HOST_IOT_KDR2	0x1A58		This register holds bits 95:64 of K_DR. The value of this register is saved in the CRYPTOCELL AO power domain.
HOST_IOT_KDR3	0x1A5C		This register holds bits 127:96 of K_DR. The value of this register is saved in the CRYPTOCELL AO power domain.
HOST_IOT_LCS	0x1A60		Controls lifecycle state (LCS) for CRYPTOCELL subsystem

Table 31: Register overview

6.1.8.1.1.1 HOST_CRYPTOKEY_SEL

Address offset: 0x1A38

AES hardware key select

If the HOST_IOT_KPRTL_LOCK register is set, and the HOST_CRYPTOKEY_SEL register set to 1, then the HW key that is connected to the AES engine is zero

Bit number	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
ID																															A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	HOST_CRYPTOKEY_SEL			Select the source of the HW key that is used by the AES engine
			K_DR	0	Use device root key K_DR from CRYPTOCELL AO power domain
			K_PRTL	1	Use hard-coded RTL key K_PRTL
			Session	2	Use provided session key

6.1.8.1.1.2 HOST_IOT_KPRTL_LOCK

Address offset: 0x1A4C

This write-once register is the K_PRTL lock register. When this register is set, K_PRTL cannot be used and a zeroed key will be used instead. The value of this register is saved in the CRYPTOCELL AO power domain.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	HOST_IOT_KPRTL_LOCK			This register is the K_PRTL lock register. When this register is set, K_PRTL cannot be used and a zeroed key will be used instead. The value of this register is saved in the CRYPTOCELL AO power domain.
			Disabled	0	K_PRTL can be selected for use from register HOST_CRYPTOKEY_SEL
			Enabled	1	K_PRTL has been locked until next power-on reset (POR). If K_PRTL is selected anyway, a zeroed key will be used instead.

6.1.8.1.1.3 HOST_IOT_KDR0

Address offset: 0x1A50

This register holds bits 31:0 of K_DR. The value of this register is saved in the CRYPTOCELL AO power domain. Reading from this address returns the K_DR valid status indicating if K_DR is successfully retained.

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	HOST_IOT_KDR0			Write: K_DR bits 31:0. Read: 0x00000000 when 128-bit K_DR key value is not yet retained in the CRYPTOCELL AO power domain. Read: 0x00000001 when 128-bit K_DR key value is successfully retained in the CRYPTOCELL AO power domain.

6.1.8.1.1.4 HOST_IOT_KDR1

Address offset: 0x1A54

This register holds bits 63:32 of K_DR. The value of this register is saved in the CRYPTOCELL AO power domain.

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	HOST_IOT_KDR1			K_DR bits 63:32

6.1.8.1.1.5 HOST_IOT_KDR2

Address offset: 0x1A58

This register holds bits 95:64 of K_DR. The value of this register is saved in the CRYPTOCELL AO power domain.

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	HOST_IOT_KDR2			K_DR bits 95:64

6.1.8.1.1.6 HOST_IOT_KDR3

Address offset: 0x1A5C

This register holds bits 127:96 of K_DR. The value of this register is saved in the CRYPTOCELL AO power domain.

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	HOST_IOT_KDR3			K_DR bits 127:96

6.1.8.1.1.7 HOST_IOT_LCS

Address offset: 0x1A60

Controls lifecycle state (LCS) for CRYPTOCELL subsystem

Bit number	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
ID																								B						A	A	A
Reset 0x00000002	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LCS			Lifecycle state value. This field is write-once per reset.
			Debug	0	CC310 operates in debug mode
			Secure	2	CC310 operates in secure mode
B	RW	LCS_IS_VALID			Read-only field. Indicates if CRYPTOCELL LCS has been successfully configured since last reset.
			Invalid	0	Valid LCS not yet retained in the CRYPTOCELL AO power domain
			Valid	1	Valid LCS successfully retained in the CRYPTOCELL AO power domain

6.2 DPPI - Distributed programmable peripheral interconnect

The distributed programmable peripheral interconnect (DPPI) enables peripherals to interact autonomously with each other by using tasks and events, without any intervention from the CPU. DPPI allows precise synchronization between peripherals when real-time application constraints exist, and eliminates the need for CPU involvement to implement behavior which can be predefined using the DPPI.

Note: Read Peripheral interface on page 14 to get familiarized with tasks, events, publish/subscribe, interrupts and other concepts.

The DPPI has the following features:

- Peripheral tasks can subscribe to channels
- Peripheral events can be published on channels
- Publish/subscribe pattern enabling multiple connection options that include the following:
  - One-to-one
  - One-to-many
  - Many-to-one
  - Many-to-many

The DPPI consists of several PPIBus modules, which are connected to a fixed number of DPPI channels and a DPPI configuration (DPPIC).

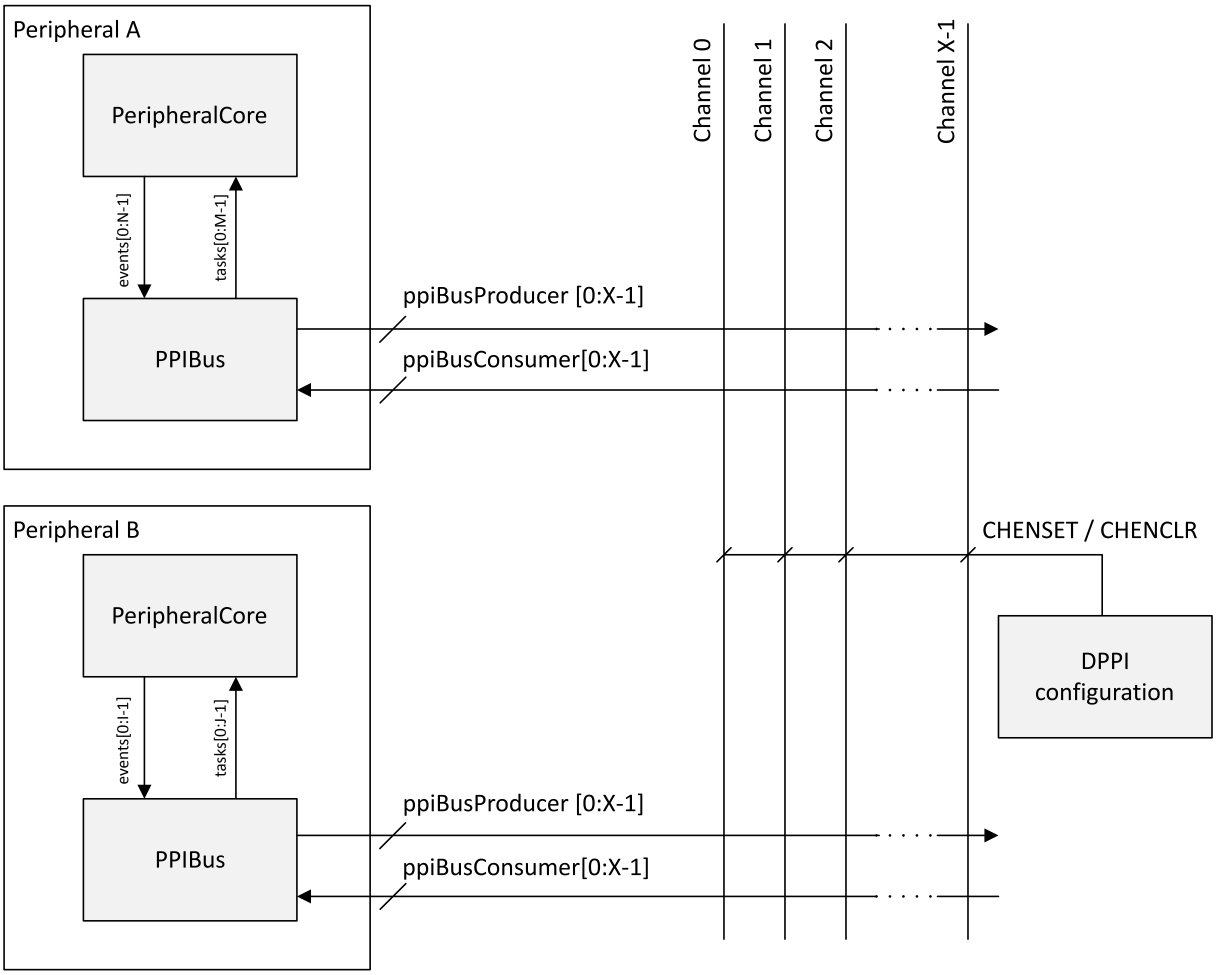

Figure 13: DPPI overview

6.2.1 Subscribing to and publishing on channels

The PPIBus can route peripheral events onto the channels (publishing), or route events from the channels into peripheral tasks (subscribing).

All peripherals include:

- One subscribe register per task
- One publish register per event

Publish and subscribe registers use a channel index field to determine the channel to which the event is published or tasks subscribed. In addition, there is an enable bit for the subscribe and publish registers that needs to be enabled before the subscription or publishing takes effect.

One event can trigger multiple tasks by subscribing different tasks to the same channel. Similarly, one task can be triggered by multiple events by publishing different events to the same channel. For advanced use cases, multiple events and multiple tasks can connect to the same channel forming a many-to-many connection. If multiple events are published on the same channel at the same time, the events are merged and only one event is routed through the DPPI.

How peripheral events are routed onto different channels based on publish registers is illustrated in the figure below.

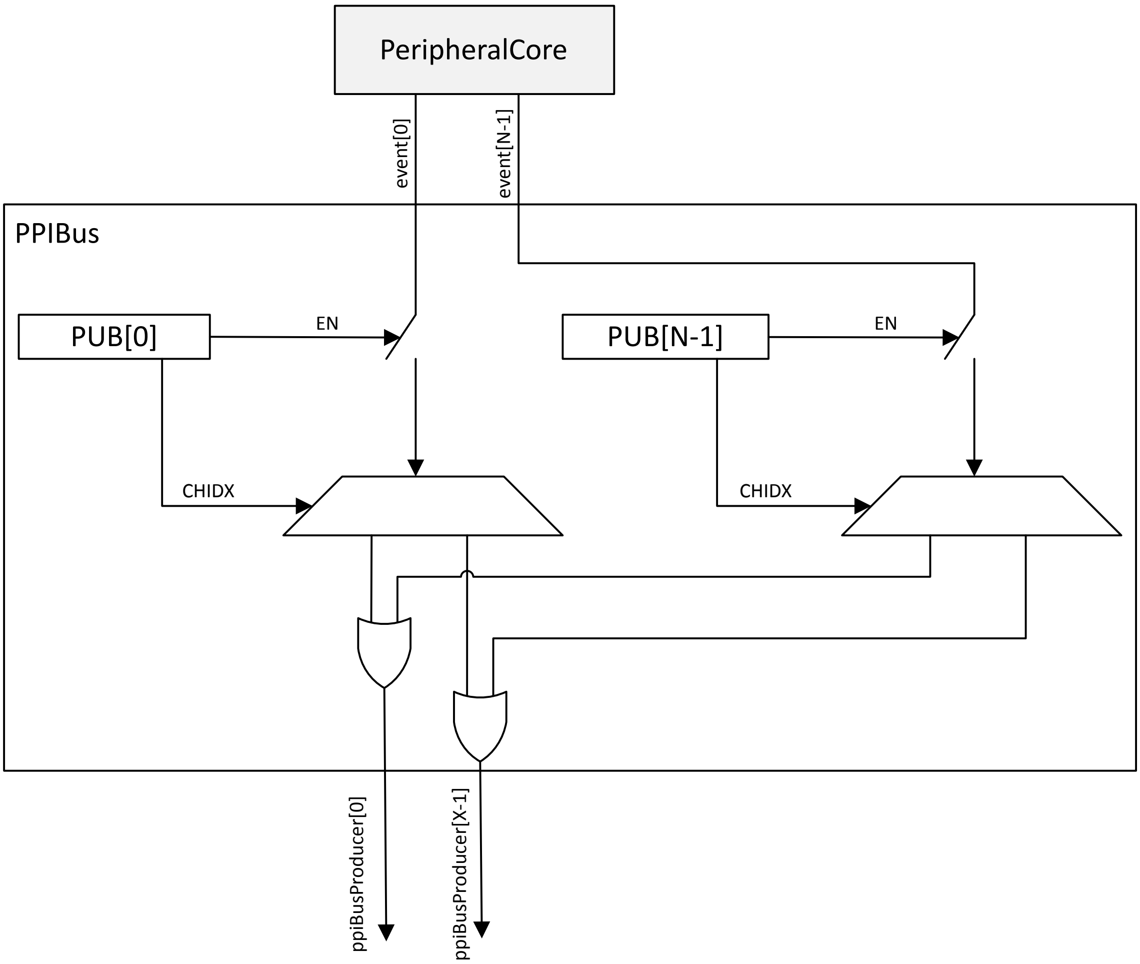

Figure 14: DPPI events flow

How peripheral tasks are triggered from different channels based on subscribe registers is illustrated in the figure below.

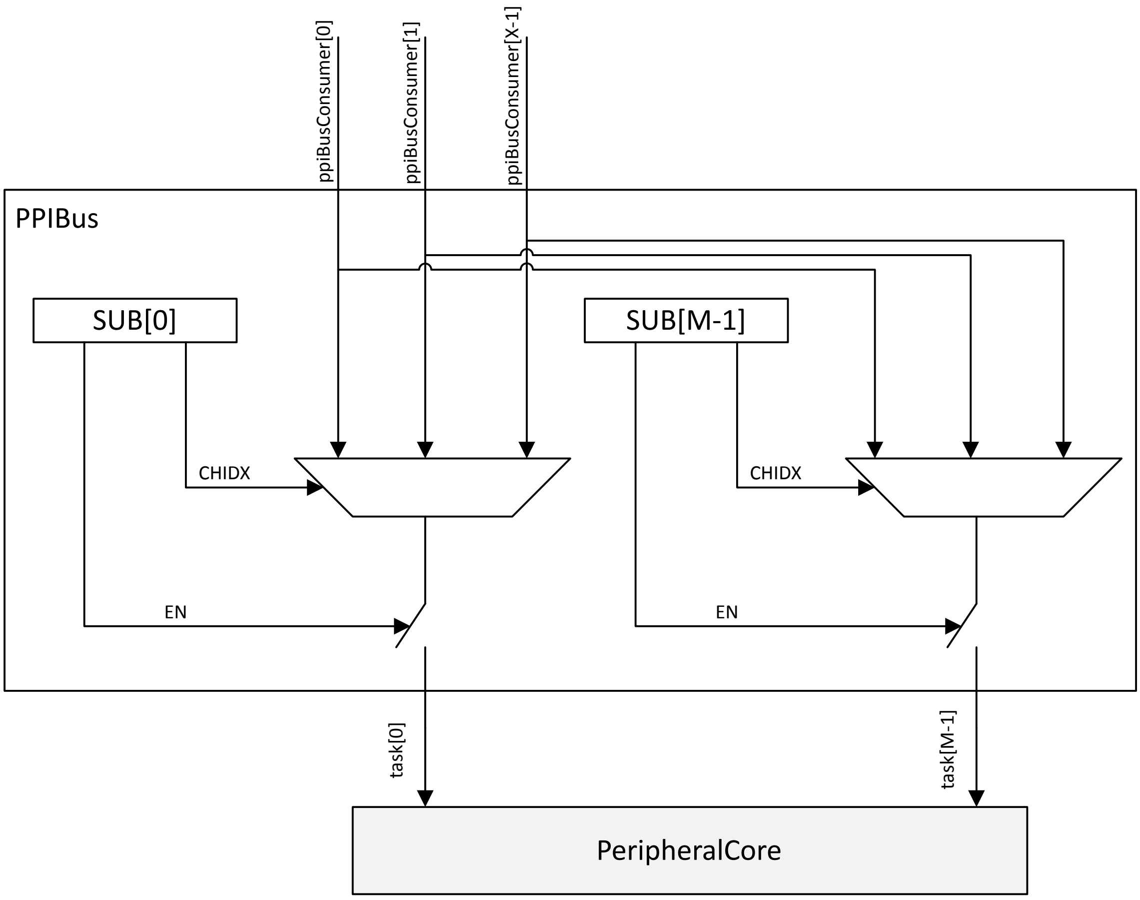

Figure 15: DPPI tasks flow

6.2.2 DPPI configuration (DPPIC)

Enabling and disabling of channels globally is handled through the DPPI configuration (DPPIC). Connection (connect/disconnect) between a channel and a peripheral is handled locally by the PPIBus.

There are two ways of enabling and disabling global channels using the DPPI configuration:

- Enable or disable channels individually using registers CHEN, CHENSET, and CHENCLR
- Enable or disable channels in channel groups using the groups' tasks ENABLE and DISABLE. It needs to be defined which channels belong to which channel groups before these tasks are triggered.

Note: ENABLE tasks are prioritized over DISABLE tasks. When a channel belongs to two or more groups, for example group m and n, and the tasks CHG[m].EN and CHG[n].DIS occur simultaneously (m and n can be equal or different), the CHG[m].EN task on that channel is prioritized.

The DPPI configuration tasks (for example CHG[0].EN) can be triggered through DPPI like any other task, which means they can be linked to a DPPI channel through the subscribe registers.

In order to write to CHG[x], the corresponding CHG[x].EN and CHG[x].DIS subscribe registers must be disabled. Writes to CHG[x] are ignored if any of the two subscribe registers are enabled.

6.2.3 Connection examples

DPPI offers several connection options. Examples are given for how to create one-to-one and many-to-many connections.

One-to-one connection

This example shows how to create a one-to-one connection between TIMER compare register and SAADC start task.

The channel configuration is set up first. TIMER0 will publish its COMPARE0 event on channel 0, and SAADC will subscribe its START task to events on the same channel. After that, the channel is enabled through the DPPIC.

```
NRF_TIMER0->PUBLISH_COMPARE0 = (DPPI_PUB_CHIDX_Ch0) |
                               (DPPI_PUB_EN_Msk);
NRF_SAADC->SUBSCRIBE_START   = (DPPI_SUB_CHIDX_Ch0) |
                               (DPPI_SUB_EN_Msk);

NRF_DPPIC->CHENSET = (DPPI_CHENSET_CH0_Set << DPPI_CHENSET_CH0_Pos);
```

Many-to-many connection

The example shows how to create a many-to-many connection, showcasing the DPPIC's channel group functionality.

A channel group that includes only channel 0 is set up first. Then the GPIOTE and TIMER0 configure their IN0 and COMPARE0 events respectively to be published on channel 0, while the SAADC configures its START task to subscribe to events on channel 0. Through DPPIC, the CHG0 DISABLE task is configured to subscribe to events on channel 0. After an event is received on channel 0 it will be disabled. Finally, channel 0 is enabled using the DPPIC task to enable a channel group.

```
NRF_DPPIC->CHG[0] = (DPPI_CHG_CH0_Included << PPI_CHG_CH0_Pos);

NRF_GPIOTE->PUBLISH_IN0          = (DPPI_PUB_CHIDX_Ch0) |
                                   (DPPI_PUB_EN_Msk);
NRF_TIMER0->PUBLISH_COMPARE0     = (DPPI_PUB_CHIDX_Ch0) |
                                   (DPPI_PUB_EN_Msk);
NRF_SAADC->SUBSCRIBE_START       = (DPPI_SUB_CHIDX_Ch0) |
                                   (DPPI_SUB_EN_Msk);
NRF_DPPIC->SUBSCRIBE_CHG[0].DIS = (DPPI_SUB_CHIDX_Ch0) |
                                   (DPPI_SUB_EN_Msk);

NRF_DPPIC->TASK_CHG[0].EN = 1;
```

6.2.4 Special considerations for a system implementing TrustZone for Cortex-M processors

DPPI is implemented with split security, meaning it handles both secure and non-secure accesses. In a system implementing the TrustZone® for Cortex®-M technology, DPPI channels can be defined as secure or non-secure using the SPU.

A peripheral configured as non-secure will only be able to subscribe to or publish on non-secure DPPI channels. A peripheral configured as secure will be able to access all DPPI channels. DPPI handles both secure and non-secure accesses, but behaves differently depending on the access type:

- A non-secure peripheral access can only configure and control the DPPI channels defined as non-secure in the SPU.DPPI.PERM[] register(s)

- A secure peripheral access can control all the DPPI channels, independently of the SPU.DPPI.PERM[] register(s)

A group of channels can be created, making it possible to simultaneously enable or disable all channels within the group. The security attribute of a group of channels (secure or non-secure) is defined as follows:

- If all channels (enabled or not) within a group are non-secure, then the group is considered non-secure
- If at least one of the channels (enabled or not) within the group is secure, then the group is considered secure

A non-secure access to a DPPI register, or a bit field, controlling a channel marked as secure in SPU.DPPI[].PERM register(s) will be ignored. Write accesses will have no effect, and read accesses will always return a zero value.

No exceptions are triggered when non-secure accesses target a register or a bit field controlling a secure channel. For example, if the bit `i` is set in the SPU.DPPI[0].PERM register (declaring DPPI channel i as secure), then:

- Non-secure write accesses to registers CHEN, CHENSET, and CHENCLR cannot write bit `i` of these registers
- Non-secure write accesses to TASK_CHG[j].EN and TASK_CHG[j].DIS registers are ignored if the channel group `j` contains at least one channel defined as secure (it can be the channel i itself or any channel declared as secure)
- Non-secure read accesses to registers CHEN, CHENSET, and CHENCLR always read 0 for the bit at position `i`

For the channel configuration registers (CHG[]), access from non-secure code is only possible if the included channels are all non-secure, whether the channels are enabled or not. If a CHG[g] register included one or more secure channel(s), then the group `g` is considered as secure, and only secure transfers can read to or write from CHG[g]. A non-secure write access is ignored, and a non-secure read access returns 0.

The DPPI can subscribe to secure and non-secure channels through the SUBSCRIBE_CHG[] registers, in order to trigger the task for enabling or disabling groups of channels. An event from a non-secure channel will be ignored if the group subscribing to this channel is secure. An event from a secure channel can trigger both secure and non-secure tasks.

6.2.5 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50017000 0x40017000	DPPIC	DPPIC : S DPPIC : NS	SPLIT	NA	DPPI configuration	

Table 32: Instances

Register	Offset	Security	Description
TASKS_CHG[0].EN	0x000		Enable channel group 0
TASKS_CHG[0].DIS	0x004		Disable channel group 0
TASKS_CHG[1].EN	0x008		Enable channel group 1
TASKS_CHG[1].DIS	0x00C		Disable channel group 1
TASKS_CHG[2].EN	0x010		Enable channel group 2
TASKS_CHG[2].DIS	0x014		Disable channel group 2
TASKS_CHG[3].EN	0x018		Enable channel group 3
TASKS_CHG[3].DIS	0x01C		Disable channel group 3
TASKS_CHG[4].EN	0x020		Enable channel group 4

Register	Offset	Security	Description
TASKS_CHG[4].DIS	0x024		Disable channel group 4
TASKS_CHG[5].EN	0x028		Enable channel group 5
TASKS_CHG[5].DIS	0x02C		Disable channel group 5
SUBSCRIBE_CHG[0].EN	0x080		Subscribe configuration for task CHG[0].EN
SUBSCRIBE_CHG[0].DIS	0x084		Subscribe configuration for task CHG[0].DIS
SUBSCRIBE_CHG[1].EN	0x088		Subscribe configuration for task CHG[1].EN
SUBSCRIBE_CHG[1].DIS	0x08C		Subscribe configuration for task CHG[1].DIS
SUBSCRIBE_CHG[2].EN	0x090		Subscribe configuration for task CHG[2].EN
SUBSCRIBE_CHG[2].DIS	0x094		Subscribe configuration for task CHG[2].DIS
SUBSCRIBE_CHG[3].EN	0x098		Subscribe configuration for task CHG[3].EN
SUBSCRIBE_CHG[3].DIS	0x09C		Subscribe configuration for task CHG[3].DIS
SUBSCRIBE_CHG[4].EN	0x0A0		Subscribe configuration for task CHG[4].EN
SUBSCRIBE_CHG[4].DIS	0x0A4		Subscribe configuration for task CHG[4].DIS
SUBSCRIBE_CHG[5].EN	0x0A8		Subscribe configuration for task CHG[5].EN
SUBSCRIBE_CHG[5].DIS	0x0AC		Subscribe configuration for task CHG[5].DIS
CHEN	0x500		Channel enable register
CHENSET	0x504		Channel enable set register
CHENCLR	0x508		Channel enable clear register
CHG[0]	0x800		Channel group 0 Note: Writes to this register are ignored if either SUBSCRIBE_CHG[0].EN or SUBSCRIBE_CHG[0].DIS is enabled
CHG[1]	0x804		Channel group 1 Note: Writes to this register are ignored if either SUBSCRIBE_CHG[1].EN or SUBSCRIBE_CHG[1].DIS is enabled
CHG[2]	0x808		Channel group 2 Note: Writes to this register are ignored if either SUBSCRIBE_CHG[2].EN or SUBSCRIBE_CHG[2].DIS is enabled
CHG[3]	0x80C		Channel group 3 Note: Writes to this register are ignored if either SUBSCRIBE_CHG[3].EN or SUBSCRIBE_CHG[3].DIS is enabled
CHG[4]	0x810		Channel group 4 Note: Writes to this register are ignored if either SUBSCRIBE_CHG[4].EN or SUBSCRIBE_CHG[4].DIS is enabled
CHG[5]	0x814		Channel group 5 Note: Writes to this register are ignored if either SUBSCRIBE_CHG[5].EN or SUBSCRIBE_CHG[5].DIS is enabled

Table 33: Register overview

6.2.5.1 TASKS_CHG[n].EN (n=0..5)

Address offset: 0x000 + (n × 0x8)

Enable channel group n

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	W	EN			Enable channel group n
			Trigger	1	Trigger task

6.2.5.2 TASKS_CHG[n].DIS (n=0..5)

Address offset: 0x004 + (n × 0x8)

Disable channel group n

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	DIS			Disable channel group n
			Trigger	1	Trigger task

6.2.5.3 SUBSCRIBE_CHG[n].EN (n=0..5)

Address offset: 0x080 + (n × 0x8)

Subscribe configuration for task CHG[n].EN

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task CHG[n].EN will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.2.5.4 SUBSCRIBE_CHG[n].DIS (n=0..5)

Address offset: 0x084 + (n × 0x8)

Subscribe configuration for task CHG[n].DIS

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task CHG[n].DIS will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.2.5.5 CHEN

Address offset: 0x500

Channel enable register

Bit number				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	
ID				P O N M L K J I H G F E D C B A	
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**	
ID	Acce	Field	Value ID	Value	Description
A-P	RW	CH[i] (i=0..15)			Enable or disable channel i
			Disabled	0	Disable channel
			Enabled	1	Enable channel

6.2.5.6 CHENSET

Address offset: 0x504

Channel enable set register

Read: Reads value of CH{i} field in CHEN register

Bit number				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	
ID				P O N M L K J I H G F E D C B A	
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**	
ID	Acce	Field	Value ID	Value	Description
A-P	RW	CH[i] (i=0..15)			Channel i enable set register. Writing 0 has no effect.
			Disabled	0	Read: Channel disabled
			Enabled	1	Read: Channel enabled
			Set	1	Write: Enable channel

6.2.5.7 CHENCLR

Address offset: 0x508

Channel enable clear register

Read: Reads value of CH{i} field in CHEN register

Bit number				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	
ID				P O N M L K J I H G F E D C B A	
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**	
ID	Acce	Field	Value ID	Value	Description
A-P	RW	CH[i] (i=0..15)			Channel i enable clear register. Writing 0 has no effect.
			Disabled	0	Read: Channel disabled
			Enabled	1	Read: Channel enabled
			Clear	1	Write: Disable channel

6.2.5.8 CHG[n] (n=0..5)

Address offset: 0x800 + (n × 0x4)

Channel group n

Note: Writes to this register are ignored if either SUBSCRIBE_CHG[n].EN or SUBSCRIBE_CHG[n].DIS is enabled

Bit number				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
ID				P O N M L K J I H G F E D C B A
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
ID	Acce Field	Value ID	Value	Description
A-P	RW CH[i] (i=0..15)			Include or exclude channel i
		Excluded	0	Exclude
		Included	1	Include

6.3 EGU — Event generator unit

Event generator unit (EGU) provides support for interlayer signaling. This means providing support for atomic triggering of both CPU execution and hardware tasks, from both firmware (by CPU) and hardware (by PPI). This feature can, for instance, be used for triggering CPU execution at a lower priority execution from a higher priority execution, or to handle a peripheral's interrupt service routine (ISR) execution at a lower priority for some of its events. However, triggering any priority from any priority is possible.

Listed here are the main EGU features:

- Software-enabled interrupt triggering
- Separate interrupt vectors for every EGU instance
- Up to 16 separate event flags per interrupt for multiplexing

Each instance of EGU implements a set of tasks which can individually be triggered to generate the corresponding event, for example, the corresponding event for TASKS_TRIGGER[n] is EVENTS_TRIGGERED[n]. See Instances on page 91 for a list of EGU instances.

6.3.1 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x5001B000 0x4001B000	EGU	EGU0 : S EGU0 : NS	US	NA	Event generator unit 0	
0x5001C000 0x4001C000	EGU	EGU1 : S EGU1 : NS	US	NA	Event generator unit 1	
0x5001D000 0x4001D000	EGU	EGU2 : S EGU2 : NS	US	NA	Event generator unit 2	
0x5001E000 0x4001E000	EGU	EGU3 : S EGU3 : NS	US	NA	Event generator unit 3	
0x5001F000 0x4001F000	EGU	EGU4 : S EGU4 : NS	US	NA	Event generator unit 4	
0x50020000 0x40020000	EGU	EGU5 : S EGU5 : NS	US	NA	Event generator unit 5	

Table 34: Instances

Register	Offset	Security	Description
TASKS_TRIGGER[0]	0x000		Trigger 0 for triggering the corresponding TRIGGERED[0] event
TASKS_TRIGGER[1]	0x004		Trigger 1 for triggering the corresponding TRIGGERED[1] event
TASKS_TRIGGER[2]	0x008		Trigger 2 for triggering the corresponding TRIGGERED[2] event
TASKS_TRIGGER[3]	0x00C		Trigger 3 for triggering the corresponding TRIGGERED[3] event
TASKS_TRIGGER[4]	0x010		Trigger 4 for triggering the corresponding TRIGGERED[4] event
TASKS_TRIGGER[5]	0x014		Trigger 5 for triggering the corresponding TRIGGERED[5] event
TASKS_TRIGGER[6]	0x018		Trigger 6 for triggering the corresponding TRIGGERED[6] event
TASKS_TRIGGER[7]	0x01C		Trigger 7 for triggering the corresponding TRIGGERED[7] event

Register	Offset	Security	Description
TASKS_TRIGGER[8]	0x020		Trigger 8 for triggering the corresponding TRIGGERED[8] event
TASKS_TRIGGER[9]	0x024		Trigger 9 for triggering the corresponding TRIGGERED[9] event
TASKS_TRIGGER[10]	0x028		Trigger 10 for triggering the corresponding TRIGGERED[10] event
TASKS_TRIGGER[11]	0x02C		Trigger 11 for triggering the corresponding TRIGGERED[11] event
TASKS_TRIGGER[12]	0x030		Trigger 12 for triggering the corresponding TRIGGERED[12] event
TASKS_TRIGGER[13]	0x034		Trigger 13 for triggering the corresponding TRIGGERED[13] event
TASKS_TRIGGER[14]	0x038		Trigger 14 for triggering the corresponding TRIGGERED[14] event
TASKS_TRIGGER[15]	0x03C		Trigger 15 for triggering the corresponding TRIGGERED[15] event
SUBSCRIBE_TRIGGER[0]	0x080		Subscribe configuration for task TRIGGER[0]
SUBSCRIBE_TRIGGER[1]	0x084		Subscribe configuration for task TRIGGER[1]
SUBSCRIBE_TRIGGER[2]	0x088		Subscribe configuration for task TRIGGER[2]
SUBSCRIBE_TRIGGER[3]	0x08C		Subscribe configuration for task TRIGGER[3]
SUBSCRIBE_TRIGGER[4]	0x090		Subscribe configuration for task TRIGGER[4]
SUBSCRIBE_TRIGGER[5]	0x094		Subscribe configuration for task TRIGGER[5]
SUBSCRIBE_TRIGGER[6]	0x098		Subscribe configuration for task TRIGGER[6]
SUBSCRIBE_TRIGGER[7]	0x09C		Subscribe configuration for task TRIGGER[7]
SUBSCRIBE_TRIGGER[8]	0x0A0		Subscribe configuration for task TRIGGER[8]
SUBSCRIBE_TRIGGER[9]	0x0A4		Subscribe configuration for task TRIGGER[9]
SUBSCRIBE_TRIGGER[10]	0x0A8		Subscribe configuration for task TRIGGER[10]
SUBSCRIBE_TRIGGER[11]	0x0AC		Subscribe configuration for task TRIGGER[11]
SUBSCRIBE_TRIGGER[12]	0x0B0		Subscribe configuration for task TRIGGER[12]
SUBSCRIBE_TRIGGER[13]	0x0B4		Subscribe configuration for task TRIGGER[13]
SUBSCRIBE_TRIGGER[14]	0x0B8		Subscribe configuration for task TRIGGER[14]
SUBSCRIBE_TRIGGER[15]	0x0BC		Subscribe configuration for task TRIGGER[15]
EVENTS_TRIGGERED[0]	0x100		Event number 0 generated by triggering the corresponding TRIGGER[0] task
EVENTS_TRIGGERED[1]	0x104		Event number 1 generated by triggering the corresponding TRIGGER[1] task
EVENTS_TRIGGERED[2]	0x108		Event number 2 generated by triggering the corresponding TRIGGER[2] task
EVENTS_TRIGGERED[3]	0x10C		Event number 3 generated by triggering the corresponding TRIGGER[3] task
EVENTS_TRIGGERED[4]	0x110		Event number 4 generated by triggering the corresponding TRIGGER[4] task
EVENTS_TRIGGERED[5]	0x114		Event number 5 generated by triggering the corresponding TRIGGER[5] task
EVENTS_TRIGGERED[6]	0x118		Event number 6 generated by triggering the corresponding TRIGGER[6] task
EVENTS_TRIGGERED[7]	0x11C		Event number 7 generated by triggering the corresponding TRIGGER[7] task
EVENTS_TRIGGERED[8]	0x120		Event number 8 generated by triggering the corresponding TRIGGER[8] task
EVENTS_TRIGGERED[9]	0x124		Event number 9 generated by triggering the corresponding TRIGGER[9] task
EVENTS_TRIGGERED[10]	0x128		Event number 10 generated by triggering the corresponding TRIGGER[10] task
EVENTS_TRIGGERED[11]	0x12C		Event number 11 generated by triggering the corresponding TRIGGER[11] task
EVENTS_TRIGGERED[12]	0x130		Event number 12 generated by triggering the corresponding TRIGGER[12] task
EVENTS_TRIGGERED[13]	0x134		Event number 13 generated by triggering the corresponding TRIGGER[13] task
EVENTS_TRIGGERED[14]	0x138		Event number 14 generated by triggering the corresponding TRIGGER[14] task
EVENTS_TRIGGERED[15]	0x13C		Event number 15 generated by triggering the corresponding TRIGGER[15] task
PUBLISH_TRIGGERED[0]	0x180		Publish configuration for event TRIGGERED[0]
PUBLISH_TRIGGERED[1]	0x184		Publish configuration for event TRIGGERED[1]
PUBLISH_TRIGGERED[2]	0x188		Publish configuration for event TRIGGERED[2]
PUBLISH_TRIGGERED[3]	0x18C		Publish configuration for event TRIGGERED[3]
PUBLISH_TRIGGERED[4]	0x190		Publish configuration for event TRIGGERED[4]
PUBLISH_TRIGGERED[5]	0x194		Publish configuration for event TRIGGERED[5]
PUBLISH_TRIGGERED[6]	0x198		Publish configuration for event TRIGGERED[6]
PUBLISH_TRIGGERED[7]	0x19C		Publish configuration for event TRIGGERED[7]
PUBLISH_TRIGGERED[8]	0x1A0		Publish configuration for event TRIGGERED[8]
PUBLISH_TRIGGERED[9]	0x1A4		Publish configuration for event TRIGGERED[9]
PUBLISH_TRIGGERED[10]	0x1A8		Publish configuration for event TRIGGERED[10]
PUBLISH_TRIGGERED[11]	0x1AC		Publish configuration for event TRIGGERED[11]
PUBLISH_TRIGGERED[12]	0x1B0		Publish configuration for event TRIGGERED[12]

Register	Offset	Security	Description
PUBLISH_TRIGGERED[13]	0x1B4		Publish configuration for event TRIGGERED[13]
PUBLISH_TRIGGERED[14]	0x1B8		Publish configuration for event TRIGGERED[14]
PUBLISH_TRIGGERED[15]	0x1BC		Publish configuration for event TRIGGERED[15]
INTEN	0x300		Enable or disable interrupt
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt

Table 35: Register overview

6.3.1.1 TASKS_TRIGGER[n] (n=0..15)

Address offset: 0x000 + (n × 0x4)

Trigger n for triggering the corresponding TRIGGERED[n] event

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_TRIGGER			Trigger n for triggering the corresponding TRIGGERED[n] event
			Trigger	1	Trigger task

6.3.1.2 SUBSCRIBE_TRIGGER[n] (n=0..15)

Address offset: 0x080 + (n × 0x4)

Subscribe configuration for task TRIGGER[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task TRIGGER[n] will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.3.1.3 EVENTS_TRIGGERED[n] (n=0..15)

Address offset: 0x100 + (n × 0x4)

Event number n generated by triggering the corresponding TRIGGER[n] task

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_TRIGGERED			Event number n generated by triggering the corresponding TRIGGER[n] task
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.3.1.4 PUBLISH_TRIGGERED[n] (n=0..15)

Address offset: 0x180 + (n × 0x4)

Publish configuration for event TRIGGERED[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event TRIGGERED[n] will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.3.1.5 INTEN

Address offset: 0x300

Enable or disable interrupt

Bit number	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
ID																	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-P	RW	TRIGGERED[i] (i=0..15)			Enable or disable interrupt for event TRIGGERED[i]
			Disabled	0	Disable
			Enabled	1	Enable

6.3.1.6 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID																	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-P	RW	TRIGGERED[i] (i=0..15)			Write '1' to enable interrupt for event TRIGGERED[i]
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.3.1.7 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID	P O N M L K J I H G F E D C B A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A-P	RW	TRIGGERED[i] (i=0..15)			Write '1' to disable interrupt for event TRIGGERED[i]
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.3.2 Electrical specification

6.3.2.1 EGU Electrical Specification

Symbol	Description	Min.	Typ.	Max.	Units
$t_{EGU,EVT}$	Latency between setting an EGU event flag and the system setting an interrupt		1		cycles

6.4 GPIO — General purpose input/output

The general purpose input/output pins (GPIOs) are grouped as one or more ports with each port having up to 32 GPIOs.

The number of ports and GPIOs per port may vary with product variant and package. Refer to Registers on page 99 and Pin assignments on page 386 for more information about the number of GPIOs that are supported.

GPIO has the following user-configurable features:

- Up to 32 GPIO pins per GPIO port
- Configurable output drive strength
- Internal pull-up and pull-down resistors
- Wake-up from high or low level triggers on all pins
- Trigger interrupt on state changes on any pin
- All pins can be used by the PPI task/event system
- One or more GPIO outputs can be controlled through PPI and GPIOTE channels
- All pins can be individually mapped to interface blocks for layout flexibility
- GPIO state changes captured on SENSE signal can be stored by LATCH register
- Support for secure and non-secure attributes for pins in conjunction with the system protection unit (SPU - System protection unit on page 260)

GPIO port and the GPIO pin details on page 96 illustrates the GPIO port containing 32 individual pins, where `PIN0` is illustrated in more detail as a reference. All signals on the left side in the illustration are used by other peripherals in the system and therefore not directly available to the CPU.

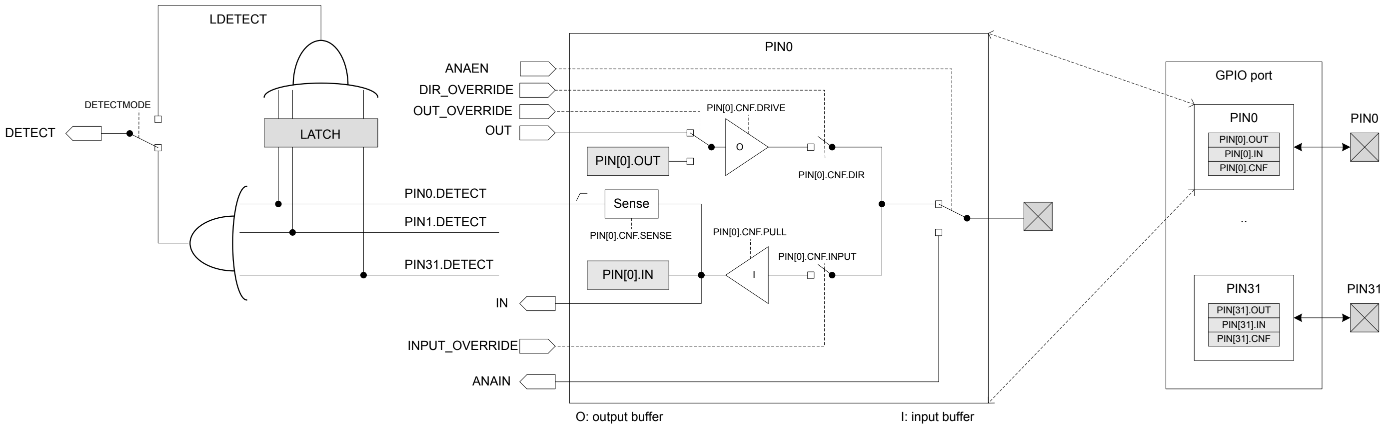

Figure 16: GPIO port and the GPIO pin details

6.4.1 Pin configuration

The GPIO port peripheral implements up to 32 pins, `PIN0` through `PIN31`. Each of these pins can be individually configured in the PIN_CNF[n] registers (n=0..31).

The following parameters can be configured through these registers:

- Direction
- Drive strength
- Enabling of pull-up and pull-down resistors
- Pin sensing
- Input buffer disconnect
- Analog input (for selected pins)

> **Note:** All write-capable registers are retained registers. See POWER — Power control on page 59 chapter for more information about retained registers.

The input buffer of a GPIO pin can be disconnected from the pin to enable power savings when the pin is not used as an input, see GPIO port and the GPIO pin details on page 96. Inputs must be connected to get a valid input value in the IN register, and for the sense mechanism to get access to the pin.

Other peripherals in the system can connect to GPIO pins and override their output value and configuration, or read their analog or digital input value. See GPIO port and the GPIO pin details on page 96.

Selected pins also support analog input signals, see ANAIN in GPIO port and the GPIO pin details on page 96. The assignment of the analog pins can be found in Pin assignments on page 386.

The following delays should be taken into considerations:

- There is 2 CPU clock cycles delay from the GPIO pad to the GPIO.IN register
- The GPIO pad must be low (or high depending on the SENSE polarity) for 3 CPU clock cycles after DETECT has gone high in order to generate a new DETECT signal

> **Note:** When a pin is configured as digital input, care has been taken to minimize increased current consumption when the input voltage is between V_{IL} and V_{IH}. However, it is a good practice to ensure that the external circuitry does not drive that pin to levels between V_{IL} and V_{IH} for a long period of time.

6.4.2 Pin sense mechanism

Pins sensitivity can be individually configured, through the SENSE field in the PIN_CNF[n] register, to detect either a high level or a low level on their input.

When the correct level is detected on any such configured pin, the sense mechanism will set the DETECT signal high. Each pin has a separate DETECT signal. Default behavior, defined by the DETECTMODE register, is that the DETECT signals from all pins in the GPIO port are combined into one common DETECT signal that is routed throughout the system, which then can be utilized by other peripherals. This mechanism is functional in both System ON and System OFF modes.

DETECTMODE and DETECTMODE_SEC are provided to handle secure and non-secure pins. DETECTMODE_SEC register is available to control the behavior associated to pin marked as secure, while the DETECTMODE register is restricted to pin marked as non-secure. Please refer to GPIO security on page 98 for more details.

Make sure that a pin is in a level that cannot trigger the sense mechanism before enabling it. The DETECT signal will go high immediately if the SENSE condition configured in the PIN_CNF registers is met when the sense mechanism is enabled. This will trigger a PORT event if the DETECT signal was low before enabling the sense mechanism.

The DETECT signal is also used by power and clock management system to exit from System OFF mode, and by GPIOTE to generate the PORT event. In addition GPIOTE_SEC is used for PORT event related to secure pins). See POWER — Power control on page 59 and GPIOTE — GPIO tasks and events on page 105 for more information about how the DETECT signal is used.

When a pin's PINx.DETECT signal goes high, a flag will be set in the LATCH register. For example, when the PIN0.DETECT signal goes high, bit 0 in the LATCH register will be set to '1'. If the CPU performs a clear operation on a bit in the LATCH register when the associated PINx.DETECT signal is high, the bit in the LATCH register will not be cleared. The LATCH register will only be cleared if the CPU explicitly clears it by writing a '1' to the bit that shall be cleared, i.e. the LATCH register will not be affected by a PINx.DETECT signal being set low.

The LDETECT signal will be set high when one or more bits in the LATCH register are '1'. The LDETECT signal will be set low when all bits in the LATCH register are successfully cleared to '0'.

If one or more bits in the LATCH register are '1' after the CPU has performed a clear operation on the LATCH registers, a rising edge will be generated on the LDETECT signal. This is illustrated in DETECT signal behavior on page 98.

> **Note:** The CPU can read the LATCH register at any time to check if a SENSE condition has been met on one or more of the the GPIO pins, even if that condition is no longer met at the time the CPU queries the LATCH register. This mechanism will work even if the LDETECT signal is not used as the DETECT signal.

The LDETECT signal is by default not connected to the GPIO port's DETECT signal, but via the DETECTMODE register it is possible to change from default behavior to DETECT signal being derived directly from the LDETECT signal instead. See GPIO port and the GPIO pin details on page 96. DETECT signal behavior on page 98 illustrates the DETECT signal behavior for these two alternatives.

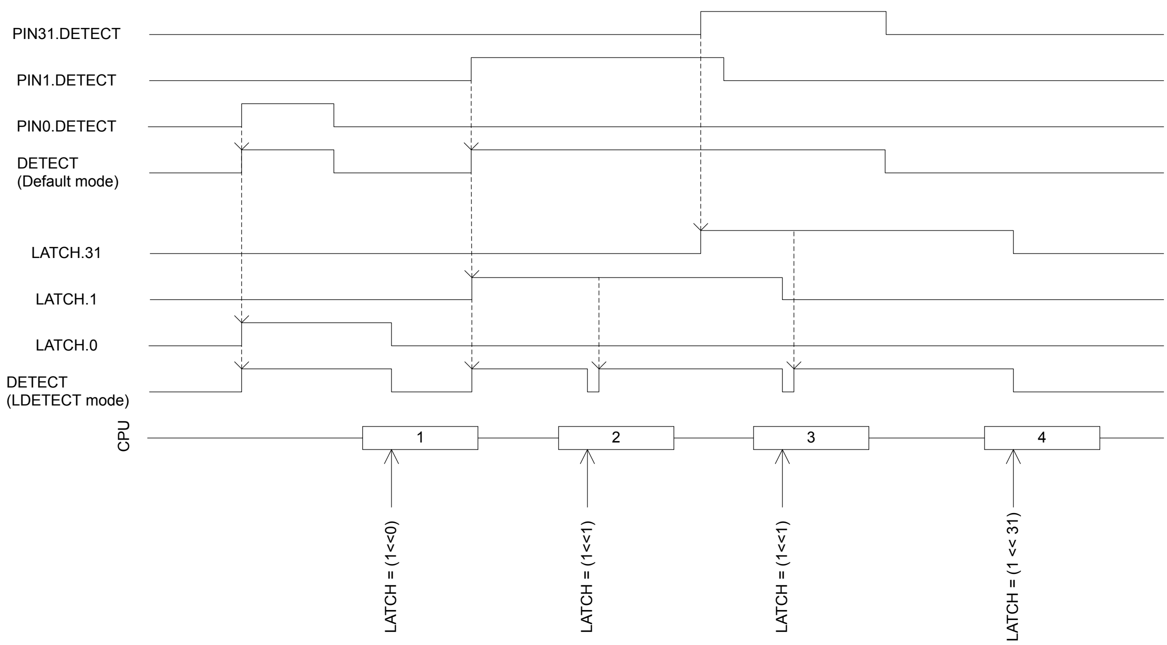

Figure 17: DETECT signal behavior

6.4.3 GPIO security

The general purpose input/output peripheral (GPIO) is implemented as a "split-security" peripheral. If marked as non-secure, it can be access by both secure and non-secure accesses but will behave differently depending of the access type :

A non-secure peripheral access will only be able to configure and control pins defined as non-secure in the system protection unit (SPU) GPIOPORT.PERM[] register(s).

A non-secure access to a register or a bitfield controlling a pin marked as secure in GPIO.PERM[] register(s) will be ignored:

- write accesses will have no effect
- read accesses will always return a zero value

No exception is triggered when a non-secure access targets a register or bitfield controlling a secure pin.

For example, if the bit *i* is set in the SPU.GPIO.PERM[0] register (declaring Pin P0.*i* as secure), then

- non-secure write accesses to OUT, OUTSET, OUTCLR, DIR, DIRSET, DIRCLR and LATCH registers will not be able to write to bit *i* of those registers
- non-secure write accesses to registers PIN[*i*].OUT and PIN_CNF[*i*] will be ignored
- non-secure read accesses to registers OUT, OUTSET, OUTCLR, IN, DIR, DIRSET, DIRCLR and LATCH will always read a 0 for the bit at position *i*
- non-secure read accesses to registers PIN[*i*].OUT, PIN[*i*].OUT and PIN_CNF[*i*] will always return 0

The GPIO.DETECTMODE and GPIO.DETECTMODE_SEC registers are handled differently than the other registers mentioned before. When accessed by a secure access, the DETECTMODE_SEC register control the source for the DETECT_SEC signal for the pins marked as secure. When accessed by a non-secure access, the DETECTMODE_SEC is read as zero and write accesses are ignored. The GPIO.DETECTMODE register controls the source for the DETECT_NSEC signal for the pins defined as non-secure.

The DETECT_NSEC signal is routed to the GPIOTE peripheral, allowing generation of events and interrupts from pins marked as non-secure. The DETECT_SEC signal is routed to the GPIOTESEC peripheral, allowing generation of events and interrupts from pins marked as secure. Principle of direct pin access on page 99 illustrates how the DETECT_NSEC and DETECT_SEC signals are generated from the GPIO PIN[].DETECT signals.

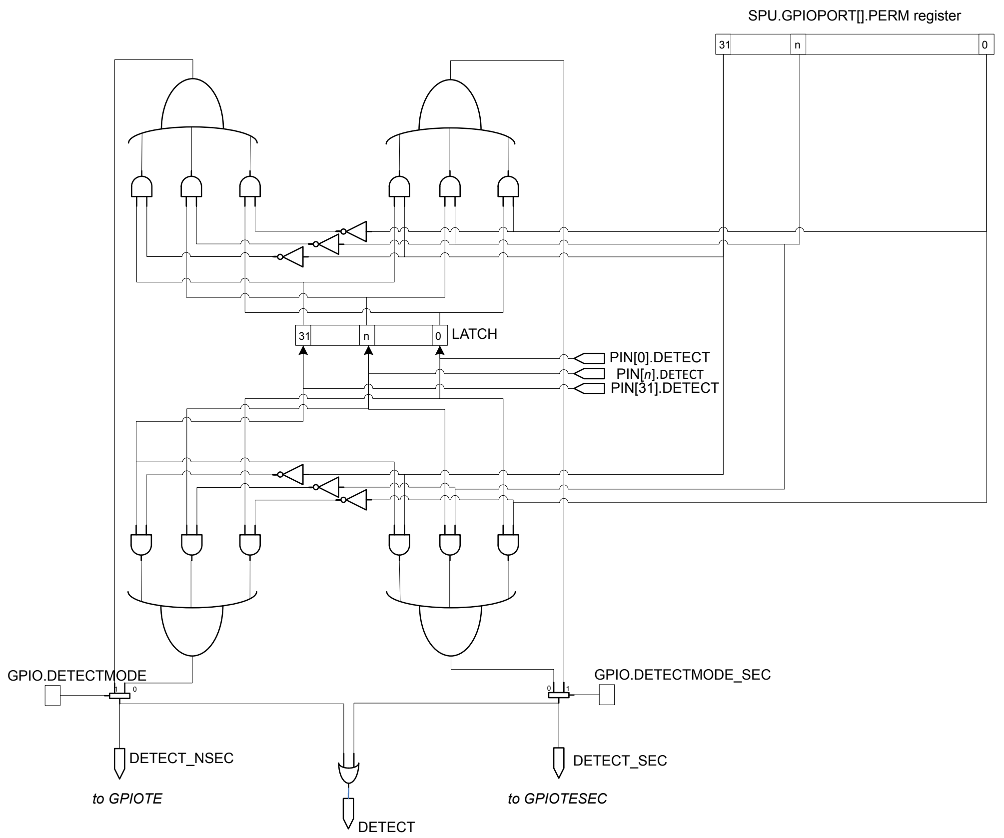

Figure 18: Principle of direct pin access

6.4.4 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50842500 0x40842500	GPIO	P0 : S P0 : NS	SPLIT	NA	General purpose input and output	

Table 36: Instances

Register	Offset	Security	Description	
OUT	0x004		Write GPIO port	Retained
OUTSET	0x008		Set individual bits in GPIO port	
OUTCLR	0x00C		Clear individual bits in GPIO port	
IN	0x010		Read GPIO port	
DIR	0x014		Direction of GPIO pins	Retained
DIRSET	0x018		DIR set register	
DIRCLR	0x01C		DIR clear register	
LATCH	0x020		Latch register indicating what GPIO pins that have met the criteria set in the PIN_CNF[n].SENSE registers	Retained
DETECTMODE	0x024		Select between default DETECT signal behavior and LDETECT mode (For non-secure pin only)	Retained

Register	Offset	Security	Description	
DETECTMODE_SEC	0x028		Select between default DETECT signal behavior and LDETECT mode (For secure pin only)	Retained
PIN_CNF[n]	0x200		Configuration of GPIO pins	Retained

Table 37: Register overview

6.4.4.1 OUT (Retained)

Address offset: 0x004

This register is a retained register

Write GPIO port

Bit number	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
ID	f	e	d	c	b	a	Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-f	RW	PIN[i] (i=0..31)			Pin i
			Low	0	Pin driver is low
			High	1	Pin driver is high

6.4.4.2 OUTSET

Address offset: 0x008

Set individual bits in GPIO port

Read: reads value of OUT register.

Bit number	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
ID	f	e	d	c	b	a	Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-f	RW	PIN[i] (i=0..31)			Pin i
			Low	0	Read: pin driver is low
			High	1	Read: pin driver is high
			Set	1	Write: writing a '1' sets the pin high; writing a '0' has no effect

6.4.4.3 OUTCLR

Address offset: 0x00C

Clear individual bits in GPIO port

Read: reads value of OUT register.

Bit number	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
ID	f	e	d	c	b	a	Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-f	RW	PIN[i] (i=0..31)			Pin i
			Low	0	Read: pin driver is low
			High	1	Read: pin driver is high
			Clear	1	Write: writing a '1' sets the pin low; writing a '0' has no effect

6.4.4.4 IN

Address offset: 0x010

Read GPIO port

Bit number	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
ID	f	e	d	c	b	a	Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-f	R	PIN[i] (i=0..31)			Pin i
			Low	0	Pin input is low
			High	1	Pin input is high

6.4.4.5 DIR (Retained)

Address offset: 0x014

This register is a retained register

Direction of GPIO pins

Bit number	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
ID	f	e	d	c	b	a	Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-f	RW	PIN[i] (i=0..31)			Pin i
			Input	0	Pin set as input
			Output	1	Pin set as output

6.4.4.6 DIRSET

Address offset: 0x018

DIR set register

Read: reads value of DIR register.

Bit number	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
ID	f	e	d	c	b	a	Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-f	RW	PIN[i] (i=0..31)			Set as output pin i
			Input	0	Read: pin set as input
			Output	1	Read: pin set as output
			Set	1	Write: writing a '1' sets pin to output; writing a '0' has no effect

6.4.4.7 DIRCLR

Address offset: 0x01C

DIR clear register

Read: reads value of DIR register.

Bit number	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
ID	f	e	d	c	b	a	Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-f	RW	PIN[i] (i=0..31)			Set as input pin i
			Input	0	Read: pin set as input
			Output	1	Read: pin set as output
			Clear	1	Write: writing a '1' sets pin to input; writing a '0' has no effect

6.4.4.8 LATCH (Retained)

Address offset: 0x020

This register is a retained register

Latch register indicating what GPIO pins that have met the criteria set in the PIN_CNF[n].SENSE registers

Bit number	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
ID	f	e	d	c	b	a	Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-f	RW	PIN[i] (i=0..31)			Status on whether PINi has met criteria set in PIN_CNFi.SENSE register. Write '1' to clear.
			NotLatched	0	Criteria has not been met
			Latched	1	Criteria has been met

6.4.4.9 DETECTMODE (Retained)

Address offset: 0x024

This register is a retained register

Select between default DETECT signal behavior and LDETECT mode (For non-secure pin only)

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	DETECTMODE			Select between default DETECT signal behavior and LDETECT mode
			Default	0	DETECT directly connected to PIN DETECT signals
			LDETECT	1	Use the latched LDETECT behavior

6.4.4.10 DETECTMODE_SEC (Retained)

Address offset: 0x028

This register is a retained register

Select between default DETECT signal behavior and LDETECT mode (For secure pin only)

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	DETECTMODE			Select between default DETECT signal behavior and LDETECT mode
			Default	0	DETECT directly connected to PIN DETECT signals
			LDETECT	1	Use the latched LDETECT behavior

6.4.4.11 PIN_CNF[n] (n=0..31) (Retained)

Address offset: 0x200 + (n × 0x4)

This register is a retained register

Configuration of GPIO pins

Bit number	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
ID	E E (bits 17–16), D D D (bits 10–8), C C (bits 3–2), B (bit 1), A (bit 0)
Reset 0x00000002	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	DIR			Pin direction. Same physical register as DIR register
			Input	0	Configure pin as an input pin
			Output	1	Configure pin as an output pin
B	RW	INPUT			Connect or disconnect input buffer
			Connect	0	Connect input buffer
			Disconnect	1	Disconnect input buffer
C	RW	PULL			Pull configuration
			Disabled	0	No pull
			Pulldown	1	Pull down on pin
			Pullup	3	Pull up on pin
D	RW	DRIVE			Drive configuration
			S0S1	0	Standard '0', standard '1'
			H0S1	1	High drive '0', standard '1'
			S0H1	2	Standard '0', high drive '1'
			H0H1	3	High drive '0', high 'drive '1''
			D0S1	4	Disconnect '0', standard '1' (normally used for wired-or connections)

Bit number	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
ID															E	E						D	D	D					C	C	B	A
Reset 0x00000002	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**

ID	Acce	Field	Value ID	Value	Description
			D0H1	5	Disconnect '0', high drive '1' (normally used for wired-or connections)
			S0D1	6	Standard '0', disconnect '1' (normally used for wired-and connections)
			H0D1	7	High drive '0', disconnect '1' (normally used for wired-and connections)
E	RW	SENSE			Pin sensing mechanism
			Disabled	0	Disabled
			High	2	Sense for high level
			Low	3	Sense for low level

6.4.5 Electrical specification

6.4.5.1 GPIO Electrical Specification

Symbol	Description	Min.	Typ.	Max.	Units
V_{IH}	Input high voltage	0.7 x VDD		VDD	V
V_{IL}	Input low voltage	VSS		0.3 x VDD	V
$V_{OH,SD}$	Output high voltage, standard drive, 0.5 mA, VDD ≥1.7	VDD-0.4		VDD	V
$V_{OH,HDH}$	Output high voltage, high drive, 5 mA, VDD >= 2.7 V	VDD-0.4		VDD	V
$V_{OH,HDL}$	Output high voltage, high drive, 3 mA, VDD >= 1.7 V	VDD-0.4		VDD	V
$V_{OL,SD}$	Output low voltage, standard drive, 0.5 mA, VDD ≥1.7	VSS		VSS+0.4	V
$V_{OL,HDH}$	Output low voltage, high drive, 5 mA, VDD >= 2.7 V	VSS		VSS+0.4	V
$V_{OL,HDL}$	Output low voltage, high drive, 3 mA, VDD >= 1.7 V	VSS		VSS+0.4	V
$I_{OL,SD}$	Current at VSS+0.4 V, output set low, standard drive, VDD ≥1.7	1	2	4	mA
$I_{OL,HDH}$	Current at VSS+0.4 V, output set low, high drive, VDD >= 2.7 V	6	10	15	mA
$I_{OL,HDL}$	Current at VSS+0.4 V, output set low, high drive, VDD >= 1.7 V	3			mA
$I_{OH,SD}$	Current at VDD-0.4 V, output set high, standard drive, VDD ≥1.7	1	2	4	mA
$I_{OH,HDH}$	Current at VDD-0.4 V, output set high, high drive, VDD >= 2.7 V	6	9	14	mA
$I_{OH,HDL}$	Current at VDD-0.4 V, output set high, high drive, VDD >= 1.7 V	3			mA
$t_{RF,15pF}$	Rise/fall time, standard drive mode, 10-90%, 15 pF load[1]	6	9	19	ns
$t_{RF,25pF}$	Rise/fall time, standard drive mode, 10-90%, 25 pF load[1]	10	13	30	ns
$t_{RF,50pF}$	Rise/fall time, standard drive mode, 10-90%, 50 pF load[1]	18	25	61	ns
$t_{HRF,15pF}$	Rise/Fall time, high drive mode, 10-90%, 15 pF load[1]	2	4	8	ns
$t_{HRF,25pF}$	Rise/Fall time, high drive mode, 10-90%, 25 pF load[1]	3	5	11	ns
$t_{HRF,50pF}$	Rise/Fall time, high drive mode, 10-90%, 50 pF load[1]	5	8	19	ns
R_{PU}	Pull-up resistance	11	13	16	kΩ
R_{PD}	Pull-down resistance	11	13	16	kΩ

[1] Rise and fall times based on simulations

Symbol	Description	Min.	Typ.	Max.	Units
C_{PAD}	Pad capacitance		3		pF

6.5 GPIOTE — GPIO tasks and events

The GPIO tasks and events (GPIOTE) module provides functionality for accessing GPIO pins using tasks and events. Each GPIOTE channel can be assigned to one pin.

A GPIOTE block enables GPIOs to generate events on pin state change which can be used to carry out tasks through the PPI system. A GPIO can also be driven to change state on system events using the PPI system. Taks and events are briefly introduced as part of the Peripheral interface on page 14, whereas GPIO is described in more detail in GPIO — General purpose input/output on page 95.

Low power detection of pin state changes is possible when in System ON or System OFF.

Instance	Number of GPIOTE channels
GPIOTE	8

Table 38: GPIOTE properties

Up to three tasks can be used in each GPIOTE channel for performing write operations to a pin. Two tasks are fixed (SET and CLR), and one (OUT) is configurable to perform following operations:

- Set
- Clear
- Toggle

An event can be generated in each GPIOTE channel from one of the following input conditions:

- Rising edge
- Falling edge
- Any change

6.5.1 Pin events and tasks

The GPIOTE module has a number of tasks and events that can be configured to operate on individual GPIO pins.

The tasks (SET[n], CLR[n] and OUT[n]) can be used for writing to individual pins, and the events (IN[n]) can be generated from changes occurring at the inputs of individual pins.

The SET task will set the pin selected in GPIOTE.CONFIG[n].PSEL to high.

The CLR task will set the pin low.

The effect of the OUT task on the pin is configurable in CONFIG[n].POLARITY , and can either set the pin high, set it low, or toggle it.

The tasks and events are configured using the CONFIG[n] registers. Every set of SET, CLR and OUT[n] tasks and IN[n] events has one CONFIG[n] register associated with it.

As long as a SET[n], CLR[n] and OUT[n] task or an IN[n] event is configured to control a pin **n**, the pin's output value will only be updated by the GPIOTE module. The pin's output value as specified in the GPIO will therefore be ignored as long as the pin is controlled by GPIOTE. Attempting to write a pin as a normal GPIO pin will have no effect. When the GPIOTE is disconnected from a pin, see MODE field in CONFIG[n] register, the associated pin will get the output and configuration values specified in the GPIO module.

When conflicting tasks are triggered simultaneously (i.e. during the same clock cycle) in one channel, the precedence of the tasks will be as described in Task priorities on page 106.

Priority	Task
1	OUT
2	CLR
3	SET

Table 39: Task priorities

When setting the CONFIG[n] registers, MODE=Disabled does not have the same effect as MODE=Task and POLARITY=None. In the latter case, a CLR or SET task occurring at the exact same time as OUT will end up with no change on the pin, according to the priorities described in the table above.

When a GPIOTE channel is configured to operate on a pin as a task, the initial value of that pin is configured in the OUTINIT field of CONFIG[n].

6.5.2 Port event

PORT is an event that can be generated from multiple input pins using the GPIO DETECT signal.

The event will be generated on the rising edge of the DETECT signal. See GPIO — General purpose input/output on page 95 for more information about the DETECT signal.

Putting the system into System ON IDLE while DETECT is high will not cause DETECT to wake the system up again. Make sure to clear all DETECT sources before entering sleep. If the LATCH register is used as a source, if any bit in LATCH is still high after clearing all or part of the register (for instance due to one of the PINx.DETECT signal still high), a new rising edge will be generated on DETECT, see Pin sense mechanism on page 97.

Trying to put the system to System OFF while DETECT is high will cause a wakeup from System OFF reset.

This feature is always enabled although the peripheral itself appears to be IDLE, that is, no clocks or other power intensive infrastructure have to be requested to keep this feature enabled. This feature can therefore be used to wake up the CPU from a WFI or WFE type sleep in System ON with all peripherals and the CPU idle, that is, lowest power consumption in System ON mode.

In order to prevent spurious interrupts from the PORT event while configuring the sources, the user shall first disable interrupts on the PORT event (through INTENCLR.PORT), then configure the sources (PIN_CNF[n].SENSE), clear any potential event that could have occurred during configuration (write '1' to EVENTS_PORT), and finally enable interrupts (through INTENSET.PORT).

6.5.3 Tasks and events pin configuration

Each GPIOTE channel is associated with one physical GPIO pin through the CONFIG.PSEL field.

When Event mode is selected in CONFIG.MODE, the pin specified by CONFIG.PSEL will be configured as an input, overriding the DIR setting in GPIO. Similarly, when Task mode is selected in CONFIG.MODE, the pin specified by CONFIG.PSEL will be configured as an output overriding the DIR setting and OUT value in GPIO. When Disabled is selected in CONFIG.MODE, the pin specified by CONFIG.PSEL will use its configuration from the PIN[n].CNF registers in GPIO. CONFIG.MODE must be disabled in order to be able to change the value of the PSEL field.

> **Note:** Only one GPIOTE channel can be assigned to one physical pin. Failing to do so may result in unpredictable behavior.

6.5.4 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x5000D000	GPIOTE	GPIOTE0	S	NA	Secure GPIO tasks and events	
0x40031000	GPIOTE	GPIOTE1	NS	NA	Non Secure GPIO tasks and events	

Table 40: Instances

Register	Offset	Security	Description
TASKS_OUT[0]	0x000		Task for writing to pin specified in CONFIG[0].PSEL. Action on pin is configured in CONFIG[0].POLARITY.
TASKS_OUT[1]	0x004		Task for writing to pin specified in CONFIG[1].PSEL. Action on pin is configured in CONFIG[1].POLARITY.
TASKS_OUT[2]	0x008		Task for writing to pin specified in CONFIG[2].PSEL. Action on pin is configured in CONFIG[2].POLARITY.
TASKS_OUT[3]	0x00C		Task for writing to pin specified in CONFIG[3].PSEL. Action on pin is configured in CONFIG[3].POLARITY.
TASKS_OUT[4]	0x010		Task for writing to pin specified in CONFIG[4].PSEL. Action on pin is configured in CONFIG[4].POLARITY.
TASKS_OUT[5]	0x014		Task for writing to pin specified in CONFIG[5].PSEL. Action on pin is configured in CONFIG[5].POLARITY.
TASKS_OUT[6]	0x018		Task for writing to pin specified in CONFIG[6].PSEL. Action on pin is configured in CONFIG[6].POLARITY.
TASKS_OUT[7]	0x01C		Task for writing to pin specified in CONFIG[7].PSEL. Action on pin is configured in CONFIG[7].POLARITY.
TASKS_SET[0]	0x030		Task for writing to pin specified in CONFIG[0].PSEL. Action on pin is to set it high.
TASKS_SET[1]	0x034		Task for writing to pin specified in CONFIG[1].PSEL. Action on pin is to set it high.
TASKS_SET[2]	0x038		Task for writing to pin specified in CONFIG[2].PSEL. Action on pin is to set it high.
TASKS_SET[3]	0x03C		Task for writing to pin specified in CONFIG[3].PSEL. Action on pin is to set it high.
TASKS_SET[4]	0x040		Task for writing to pin specified in CONFIG[4].PSEL. Action on pin is to set it high.
TASKS_SET[5]	0x044		Task for writing to pin specified in CONFIG[5].PSEL. Action on pin is to set it high.
TASKS_SET[6]	0x048		Task for writing to pin specified in CONFIG[6].PSEL. Action on pin is to set it high.
TASKS_SET[7]	0x04C		Task for writing to pin specified in CONFIG[7].PSEL. Action on pin is to set it high.
TASKS_CLR[0]	0x060		Task for writing to pin specified in CONFIG[0].PSEL. Action on pin is to set it low.
TASKS_CLR[1]	0x064		Task for writing to pin specified in CONFIG[1].PSEL. Action on pin is to set it low.
TASKS_CLR[2]	0x068		Task for writing to pin specified in CONFIG[2].PSEL. Action on pin is to set it low.
TASKS_CLR[3]	0x06C		Task for writing to pin specified in CONFIG[3].PSEL. Action on pin is to set it low.
TASKS_CLR[4]	0x070		Task for writing to pin specified in CONFIG[4].PSEL. Action on pin is to set it low.
TASKS_CLR[5]	0x074		Task for writing to pin specified in CONFIG[5].PSEL. Action on pin is to set it low.
TASKS_CLR[6]	0x078		Task for writing to pin specified in CONFIG[6].PSEL. Action on pin is to set it low.
TASKS_CLR[7]	0x07C		Task for writing to pin specified in CONFIG[7].PSEL. Action on pin is to set it low.
SUBSCRIBE_OUT[0]	0x080		Subscribe configuration for task OUT[0]
SUBSCRIBE_OUT[1]	0x084		Subscribe configuration for task OUT[1]
SUBSCRIBE_OUT[2]	0x088		Subscribe configuration for task OUT[2]
SUBSCRIBE_OUT[3]	0x08C		Subscribe configuration for task OUT[3]
SUBSCRIBE_OUT[4]	0x090		Subscribe configuration for task OUT[4]
SUBSCRIBE_OUT[5]	0x094		Subscribe configuration for task OUT[5]
SUBSCRIBE_OUT[6]	0x098		Subscribe configuration for task OUT[6]
SUBSCRIBE_OUT[7]	0x09C		Subscribe configuration for task OUT[7]
SUBSCRIBE_SET[0]	0x0B0		Subscribe configuration for task SET[0]
SUBSCRIBE_SET[1]	0x0B4		Subscribe configuration for task SET[1]
SUBSCRIBE_SET[2]	0x0B8		Subscribe configuration for task SET[2]

Register	Offset	Security	Description
SUBSCRIBE_SET[3]	0x0BC		Subscribe configuration for task SET[3]
SUBSCRIBE_SET[4]	0x0C0		Subscribe configuration for task SET[4]
SUBSCRIBE_SET[5]	0x0C4		Subscribe configuration for task SET[5]
SUBSCRIBE_SET[6]	0x0C8		Subscribe configuration for task SET[6]
SUBSCRIBE_SET[7]	0x0CC		Subscribe configuration for task SET[7]
SUBSCRIBE_CLR[0]	0x0E0		Subscribe configuration for task CLR[0]
SUBSCRIBE_CLR[1]	0x0E4		Subscribe configuration for task CLR[1]
SUBSCRIBE_CLR[2]	0x0E8		Subscribe configuration for task CLR[2]
SUBSCRIBE_CLR[3]	0x0EC		Subscribe configuration for task CLR[3]
SUBSCRIBE_CLR[4]	0x0F0		Subscribe configuration for task CLR[4]
SUBSCRIBE_CLR[5]	0x0F4		Subscribe configuration for task CLR[5]
SUBSCRIBE_CLR[6]	0x0F8		Subscribe configuration for task CLR[6]
SUBSCRIBE_CLR[7]	0x0FC		Subscribe configuration for task CLR[7]
EVENTS_IN[0]	0x100		Event generated from pin specified in CONFIG[0].PSEL
EVENTS_IN[1]	0x104		Event generated from pin specified in CONFIG[1].PSEL
EVENTS_IN[2]	0x108		Event generated from pin specified in CONFIG[2].PSEL
EVENTS_IN[3]	0x10C		Event generated from pin specified in CONFIG[3].PSEL
EVENTS_IN[4]	0x110		Event generated from pin specified in CONFIG[4].PSEL
EVENTS_IN[5]	0x114		Event generated from pin specified in CONFIG[5].PSEL
EVENTS_IN[6]	0x118		Event generated from pin specified in CONFIG[6].PSEL
EVENTS_IN[7]	0x11C		Event generated from pin specified in CONFIG[7].PSEL
EVENTS_PORT	0x17C		Event generated from multiple input GPIO pins with SENSE mechanism enabled
PUBLISH_IN[0]	0x180		Publish configuration for event IN[0]
PUBLISH_IN[1]	0x184		Publish configuration for event IN[1]
PUBLISH_IN[2]	0x188		Publish configuration for event IN[2]
PUBLISH_IN[3]	0x18C		Publish configuration for event IN[3]
PUBLISH_IN[4]	0x190		Publish configuration for event IN[4]
PUBLISH_IN[5]	0x194		Publish configuration for event IN[5]
PUBLISH_IN[6]	0x198		Publish configuration for event IN[6]
PUBLISH_IN[7]	0x19C		Publish configuration for event IN[7]
PUBLISH_PORT	0x1FC		Publish configuration for event PORT
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
CONFIG[0]	0x510		Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event
CONFIG[1]	0x514		Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event
CONFIG[2]	0x518		Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event
CONFIG[3]	0x51C		Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event
CONFIG[4]	0x520		Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event
CONFIG[5]	0x524		Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event
CONFIG[6]	0x528		Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event
CONFIG[7]	0x52C		Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event

Table 41: Register overview

6.5.4.1 TASKS_OUT[n] (n=0..7)

Address offset: 0x000 + (n × 0x4)

Task for writing to pin specified in CONFIG[n].PSEL. Action on pin is configured in CONFIG[n].POLARITY.

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_OUT			Task for writing to pin specified in CONFIG[n].PSEL. Action on pin is configured in CONFIG[n].POLARITY.
			Trigger	1	Trigger task

6.5.4.2 TASKS_SET[n] (n=0..7)

Address offset: 0x030 + (n × 0x4)

Task for writing to pin specified in CONFIG[n].PSEL. Action on pin is to set it high.

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_SET			Task for writing to pin specified in CONFIG[n].PSEL. Action on pin is to set it high.
			Trigger	1	Trigger task

6.5.4.3 TASKS_CLR[n] (n=0..7)

Address offset: 0x060 + (n × 0x4)

Task for writing to pin specified in CONFIG[n].PSEL. Action on pin is to set it low.

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_CLR			Task for writing to pin specified in CONFIG[n].PSEL. Action on pin is to set it low.
			Trigger	1	Trigger task

6.5.4.4 SUBSCRIBE_OUT[n] (n=0..7)

Address offset: 0x080 + (n × 0x4)

Subscribe configuration for task OUT[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task OUT[n] will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.5.4.5 SUBSCRIBE_SET[n] (n=0..7)

Address offset: 0x0B0 + (n × 0x4)

Subscribe configuration for task SET[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task SET[n] will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.5.4.6 SUBSCRIBE_CLR[n] (n=0..7)

Address offset: 0x0E0 + (n × 0x4)

Subscribe configuration for task CLR[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task CLR[n] will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.5.4.7 EVENTS_IN[n] (n=0..7)

Address offset: 0x100 + (n × 0x4)

Event generated from pin specified in CONFIG[n].PSEL

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_IN			Event generated from pin specified in CONFIG[n].PSEL
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.5.4.8 EVENTS_PORT

Address offset: 0x17C

Event generated from multiple input GPIO pins with SENSE mechanism enabled

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_PORT			Event generated from multiple input GPIO pins with SENSE mechanism enabled
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.5.4.9 PUBLISH_IN[n] (n=0..7)

Address offset: 0x180 + (n × 0x4)

Publish configuration for event IN[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event IN[n] will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.5.4.10 PUBLISH_PORT

Address offset: 0x1FC

Publish configuration for event PORT

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event PORT will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.5.4.11 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID	I																								H	G	F	E	D	C	B	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A-H	RW	IN[i] (i=0..7)			Write '1' to enable interrupt for event IN[i]
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
I	RW	PORT			Write '1' to enable interrupt for event PORT
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.5.4.12 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID	I																								H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-H	RW	IN[i] (i=0..7)			Write '1' to disable interrupt for event IN[i]
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
I	RW	PORT			Write '1' to disable interrupt for event PORT
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.5.4.13 CONFIG[n] (n=0..7)

Address offset: 0x510 + (n × 0x4)

Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event

Bit number	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
ID												E			D	D				B	B	B	B	B							A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MODE			Mode
			Disabled	0	Disabled. Pin specified by PSEL will not be acquired by the GPIOTE module.
			Event	1	Event mode The pin specified by PSEL will be configured as an input and the IN[n] event will be generated if operation specified in POLARITY occurs on the pin.
			Task	3	Task mode The GPIO specified by PSEL will be configured as an output and triggering the SET[n], CLR[n] or OUT[n] task will perform the operation specified by POLARITY on the pin. When enabled as a task the GPIOTE module will acquire the pin and the pin can no longer be written as a regular output pin from the GPIO module.
B	RW	PSEL		[0..31]	GPIO number associated with SET[n], CLR[n] and OUT[n] tasks and IN[n] event
D	RW	POLARITY			When In task mode: Operation to be performed on output when OUT[n] task is triggered. When In event mode: Operation on input that shall trigger IN[n] event.
			None	0	Task mode: No effect on pin from OUT[n] task. Event mode: no IN[n] event generated on pin activity.
			LoToHi	1	Task mode: Set pin from OUT[n] task. Event mode: Generate IN[n] event when rising edge on pin.
			HiToLo	2	Task mode: Clear pin from OUT[n] task. Event mode: Generate IN[n] event when falling edge on pin.
			Toggle	3	Task mode: Toggle pin from OUT[n]. Event mode: Generate IN[n] when any change on pin.
E	RW	OUTINIT			When in task mode: Initial value of the output when the GPIOTE channel is configured. When in event mode: No effect.
			Low	0	Task mode: Initial value of pin before task triggering is low

Bit number	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
ID												E			D	D				B	B	B	B	B							A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			High	1	Task mode: Initial value of pin before task triggering is high

6.5.5 Electrical specification

6.6 IPC — Interprocessor communication

The interprocessor communication (IPC) peripheral is used to send and receive events between MCUs in the system.

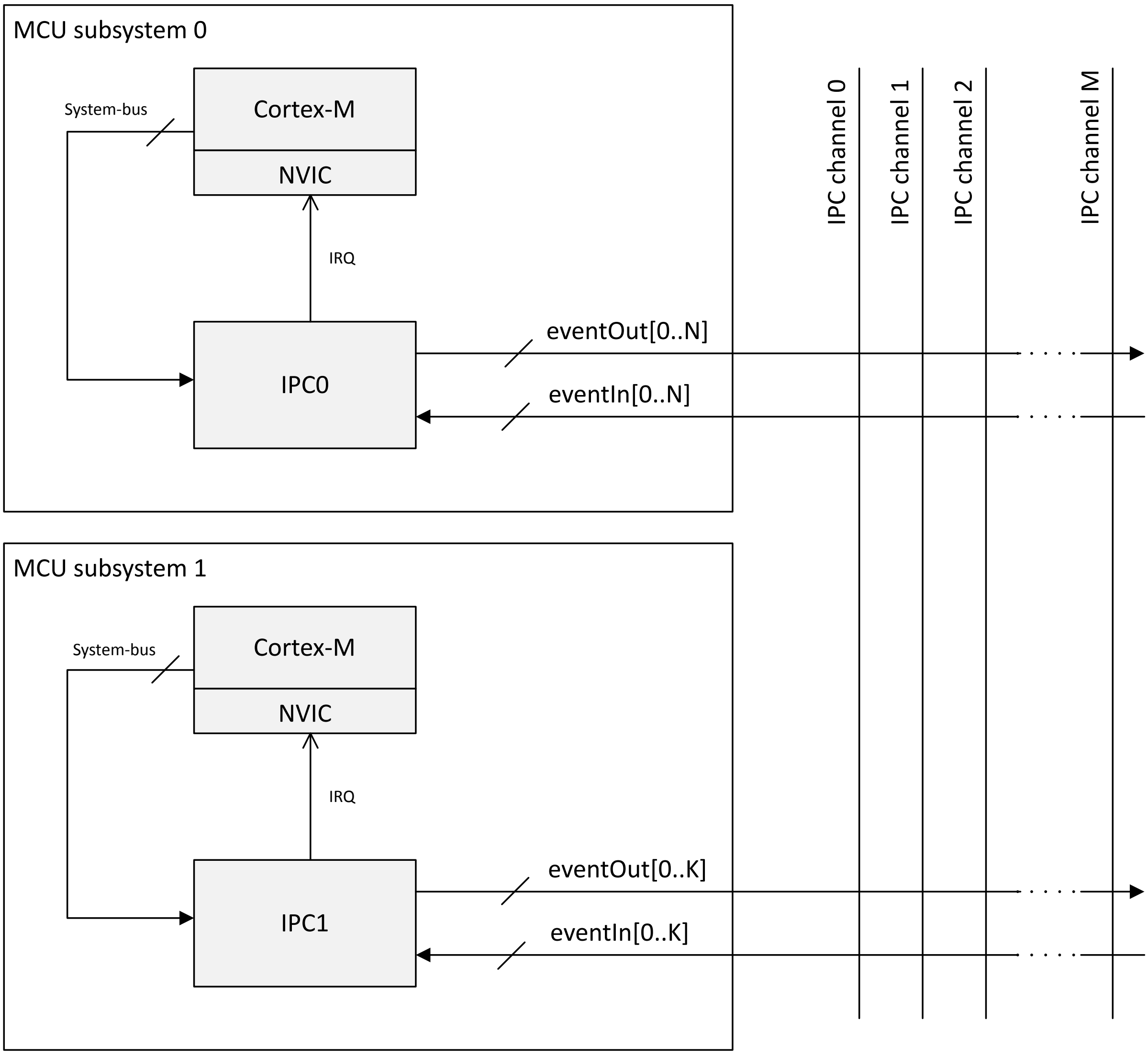

Figure 19: IPC block diagram

Functional description

IPC block diagram on page 113 illustrates the interprocessor communication (IPC) peripheral. In a multi-MCU system, each MCU has one dedicated IPC peripheral. The IPC peripheral can be used to send and receive events to and from other IPC peripherals. An instance of the IPC peripheral can have multiple SEND tasks and RECEIVE events. A single SEND task can be configured to signal an event on one or more

IPC channels, and a RECEIVE event can be configured to listen on one or more IPC channels. The IPC channels that are triggered in a SEND task can be configured through the SEND_CNF registers, and the IPC channels that trigger a RECEIVE event are configured through the RECEIVE_CNF registers. The figure below illustrates how the SEND_CNF and RECEIVE_CNF registers work. Both the SEND task and the RECEIVE event can be connected to all IPC channels.

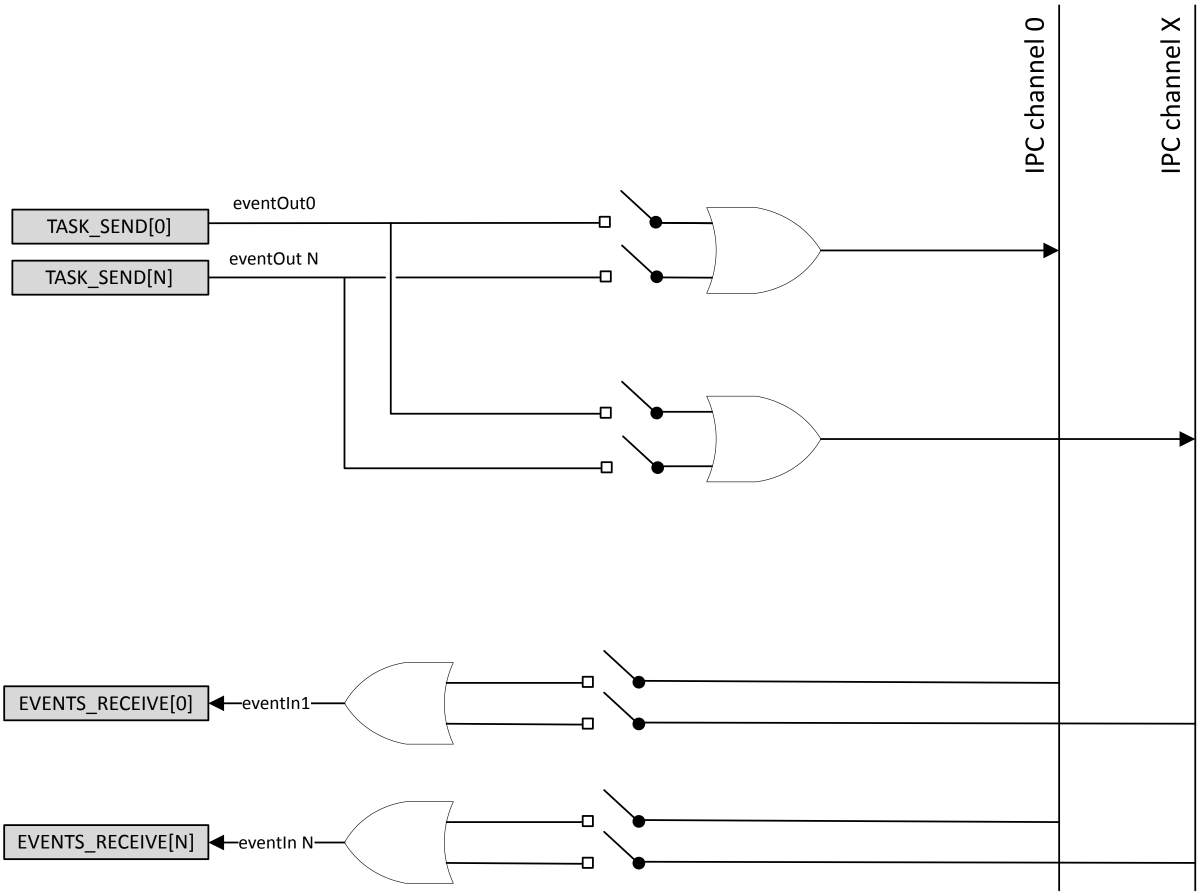

Figure 20: IPC registers SEND_CNF and RECEIVE_CNF

A SEND task can be viewed as broadcasting events onto one or more IPC channels, and a RECEIVE event can be seen as subscribing to a subset of IPC channels. It is possible for multiple IPCs to trigger events onto the same channel at the same time. When two or more events on the same channel occur within t_{IPC}, the events may be merged into a single event seen from the IPC receiver. One of the events can therefore be lost. To prevent this, the user must ensure that events on the same IPC channel do not occur within t_{IPC} of each other. When implementing firmware data structures, such as queues or mailboxes, this can be done by using one channel for acknowledgements.

An IPC event often does not contain any data itself, it is used to signal other MCUs that something has occurred. Data can be shared through shared memory, for example in the form of a software implemented mailbox, or command/event queues. It is up to software to assign a logical functionality to an IPC channel. For instance, one IPC channel can be used to signal that a command is ready to be executed, and any processor in the system can subscribe to that particular channel and decode/execute the command.

General purpose memory

The GPMEM registers can be used freely to store information. These registers are accessed like any other of the IPC peripheral's registers.

6.6.1 IPC and PPI connections

The IPC SEND tasks and RECEIVE events can be connected through PPI channels. This makes it possible to relay events from peripherals in one MCU to another, without CPU involvement.

Figure below illustrates a timer COMPARE event that is relayed from one MCU to IPC using PPI, then back into a timer CAPTURE event in another MCU.

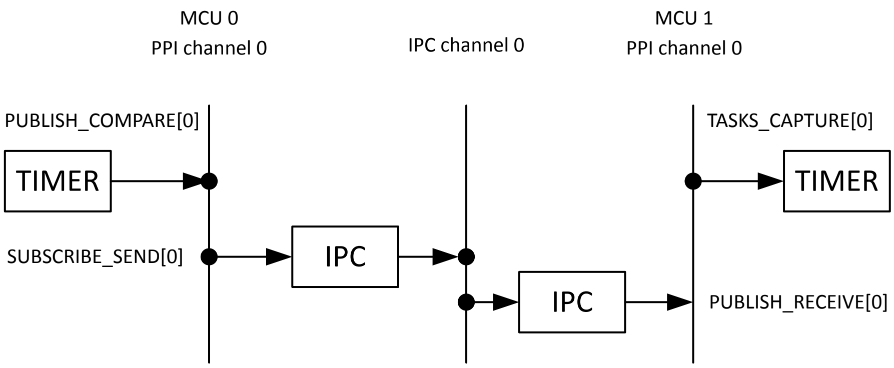

Figure 21: Example of PPI and IPC connections

6.6.2 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x5002A000 0x4002A000	IPC	IPC : S IPC : NS	US	NA	Interprocessor communication	

Table 42: Instances

Register	Offset	Security	Description
TASKS_SEND[n]	0x000		Trigger events on IPC channel enabled in SEND_CNF[n]
SUBSCRIBE_SEND[n]	0x080		Subscribe configuration for task SEND[n]
EVENTS_RECEIVE[n]	0x100		Event received on one or more of the enabled IPC channels in RECEIVE_CNF[n]
PUBLISH_RECEIVE[n]	0x180		Publish configuration for event RECEIVE[n]
INTEN	0x300		Enable or disable interrupt
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
INTPEND	0x30C		Pending interrupts
SEND_CNF[n]	0x510		Send event configuration for TASKS_SEND[n]
RECEIVE_CNF[n]	0x590		Receive event configuration for EVENTS_RECEIVE[n]
GPMEM[0]	0x610		General purpose memory
GPMEM[1]	0x614		General purpose memory
GPMEM[2]	0x618		General purpose memory
GPMEM[3]	0x61C		General purpose memory

Table 43: Register overview

6.6.2.1 TASKS_SEND[n] (n=0..7)

Address offset: 0x000 + (n × 0x4)

Trigger events on IPC channel enabled in SEND_CNF[n]

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_SEND			Trigger events on IPC channel enabled in SEND_CNF[n]
			Trigger	1	Trigger task

6.6.2.2 SUBSCRIBE_SEND[n] (n=0..7)

Address offset: 0x080 + (n × 0x4)

Subscribe configuration for task SEND[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task SEND[n] will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.6.2.3 EVENTS_RECEIVE[n] (n=0..7)

Address offset: 0x100 + (n × 0x4)

Event received on one or more of the enabled IPC channels in RECEIVE_CNF[n]

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_RECEIVE			Event received on one or more of the enabled IPC channels in RECEIVE_CNF[n]
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.6.2.4 PUBLISH_RECEIVE[n] (n=0..7)

Address offset: 0x180 + (n × 0x4)

Publish configuration for event RECEIVE[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event RECEIVE[n] will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.6.2.5 INTEN

Address offset: 0x300

Enable or disable interrupt

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-H	RW	RECEIVE[i] (i=0..7)			Enable or disable interrupt for event RECEIVE[i]
			Disabled	0	Disable
			Enabled	1	Enable

6.6.2.6 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-H	RW	RECEIVE[i] (i=0..7)			Write '1' to enable interrupt for event RECEIVE[i]
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.6.2.7 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-H	RW	RECEIVE[i] (i=0..7)			Write '1' to disable interrupt for event RECEIVE[i]
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.6.2.8 INTPEND

Address offset: 0x30C

Pending interrupts

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-H	R	RECEIVE[i] (i=0..7)			Read pending status of interrupt for event RECEIVE[i]
			NotPending	0	Read: Not pending
			Pending	1	Read: Pending

6.6.2.9 SEND_CNF[n] (n=0..7)

Address offset: 0x510 + (n × 0x4)

Send event configuration for TASKS_SEND[n]

Bit number	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
ID																									H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-H	RW	CHEN[i] (i=0..7)			Enable broadcasting on IPC channel i
			Disable	0	Disable broadcast
			Enable	1	Enable broadcast

6.6.2.10 RECEIVE_CNF[n] (n=0..7)

Address offset: 0x590 + (n × 0x4)

Receive event configuration for EVENTS_RECEIVE[n]

Bit number	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
ID																									H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-H	RW	CHEN[i] (i=0..7)			Enable subscription to IPC channel i
			Disable	0	Disable events
			Enable	1	Enable events

6.6.2.11 GPMEM[n] (n=0..3)

Address offset: 0x610 + (n × 0x4)

General purpose memory

Retained only in System ON mode

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	GPMEM			General purpose memory

6.6.3 Electrical specification

6.6.3.1 IPC Electrical Specification

Symbol	**Description**	**Min.**	**Typ.**	**Max.**	**Units**
t_{IPC}	Minimum IPC event spacing before events are merged	..	..	..	µs

6.7 I²S — Inter-IC sound interface

The I²S (Inter-IC Sound) module, supports the original two-channel I²S format, and left or right-aligned formats. It implements EasyDMA for sample transfer directly to and from RAM without CPU intervention.

The I²S peripheral has the following main features:

- Master and Slave mode
- Simultaneous bi-directional (TX and RX) audio streaming
- Original I²S and left- or right-aligned format
- 8, 16 and 24-bit sample width
- Low-jitter Master Clock generator
- Various sample rates

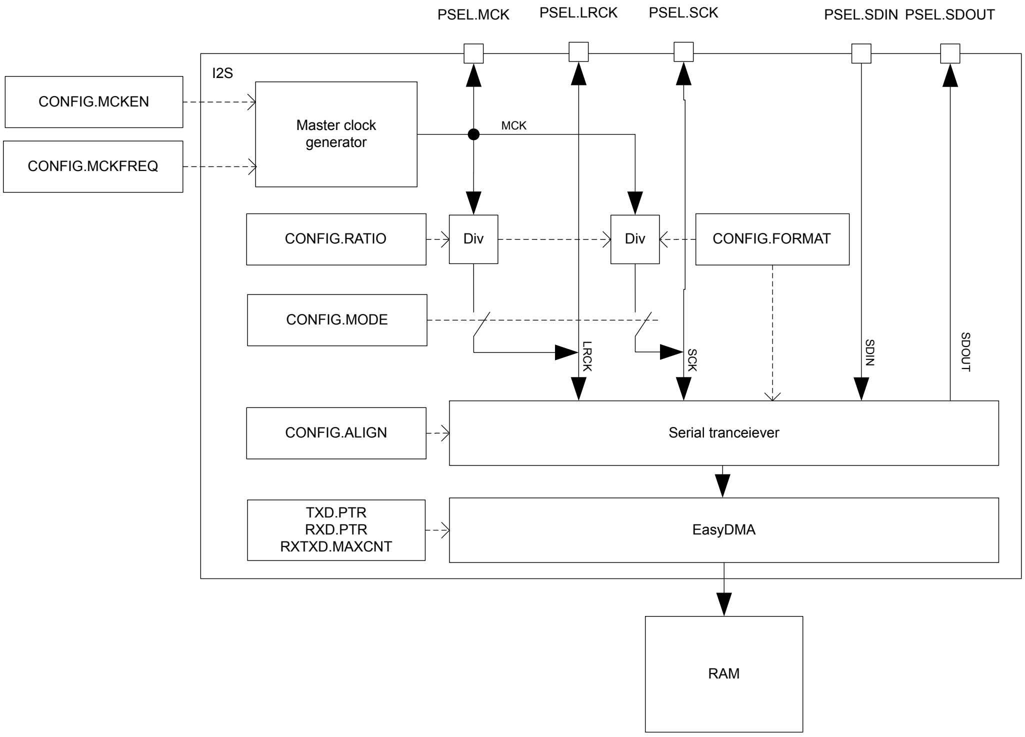

Figure 22: I²S master

6.7.1 Mode

The I²S protocol specification defines two modes of operation, Master and Slave.

The I²S mode decides which of the two sides (Master or Slave) shall provide the clock signals LRCK and SCK, and these signals are always supplied by the Master to the Slave.

6.7.2 Transmitting and receiving

The I²S module supports both transmission (TX) and reception (RX) of serial data. In both cases the serial data is shifted synchronously to the clock signals SCK and LRCK.

TX data is written to the SDOUT pin on the falling edge of SCK, and RX data is read from the SDIN pin on the rising edge of SCK. The most significant bit (MSB) is always transmitted first.

> **Note:** When starting a transmission in master mode, two frames (two left-and-right sample pairs) of value zero will be transmitted after triggering the START task, prior to the RXTXD.MAXCNT samples specified by the TXD.PTR pointer.

TX and RX are available in both Master and Slave modes and can be enabled/disabled independently in the CONFIG.TXEN on page 134 and CONFIG.RXEN on page 133.

Transmission and/or reception is started by triggering the START task. When started and transmission is enabled (in CONFIG.TXEN on page 134), the TXPTRUPD event will be generated for every RXTXD.MAXCNT on page 137 number of transmitted data words (containing one or more samples). Similarly, when started and reception is enabled (in CONFIG.RXEN on page 133), the RXPTRUPD event will be generated for every RXTXD.MAXCNT on page 137 received data words.

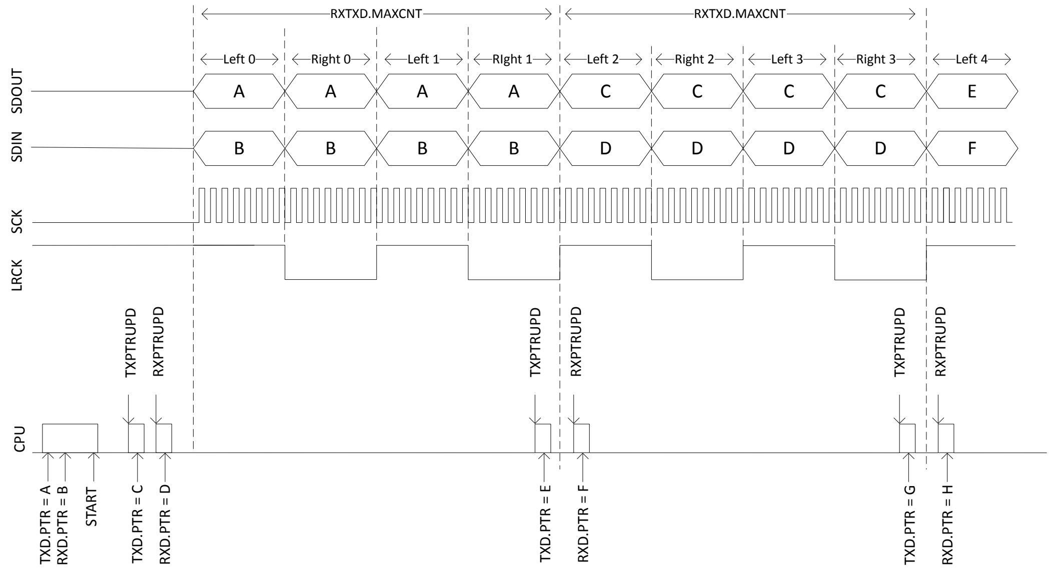

Figure 23: Transmitting and receiving. CONFIG.FORMAT = Aligned, CONFIG.SWIDTH = 8Bit, CONFIG.CHANNELS = Stereo, RXTXD.MAXCNT = 1.

6.7.3 Left right clock (LRCK)

The Left Right Clock (LRCK), often referred to as "word clock", "sample clock" or "word select" in I^2S context, is the clock defining the frames in the serial bit streams sent and received on SDOUT and SDIN, respectively.

In I2S mode, each frame contains one left and right sample pair, with the left sample being transferred during the low half period of LRCK followed by the right sample being transferred during the high period of LRCK.

In Aligned mode, each frame contains one left and right sample pair, with the left sample being transferred during the high half period of LRCK followed by the right sample being transferred during the low period of LRCK.

Consequently, the LRCK frequency is equivalent to the audio sample rate.

When operating in Master mode, the LRCK is generated from the MCK, and the frequency of LRCK is then given as:

```
LRCK = MCK / CONFIG.RATIO
```

LRCK always toggles around the falling edge of the serial clock SCK.

6.7.4 Serial clock (SCK)

The serial clock (SCK), often referred to as the serial bit clock, pulses once for each data bit being transferred on the serial data lines SDIN and SDOUT.

When operating in Master mode the SCK is generated from the MCK, and the frequency of SCK is then given as:

```
SCK = 2 * LRCK * CONFIG.SWIDTH
```

The falling edge of the SCK falls on the toggling edge of LRCK.

When operating in Slave mode SCK is provided by the external I^2S master.

6.7.5 Master clock (MCK)

The master clock (MCK) is the clock from which LRCK and SCK are derived when operating in Master mode.

The MCK is generated by an internal MCK generator. This generator always needs to be enabled when in Master mode, but the generator can also be enabled when in Slave mode. Enabling the generator when in slave mode can be useful in the case where the external Master is not able to generate its own master clock.

The MCK generator is enabled/disabled in the register CONFIG.MCKEN on page 134, and the generator is started or stopped by the START or STOP tasks.

In Master mode the LRCK and the SCK frequencies are closely related, as both are derived from MCK and set indirectly through CONFIG.RATIO on page 135 and CONFIG.SWIDTH on page 135.

When configuring these registers, the user is responsible for fulfilling the following requirements:

1. SCK frequency can never exceed the MCK frequency, which can be formulated as:

   ```
   CONFIG.RATIO >= 2 * CONFIG.SWIDTH
   ```

2. The MCK/LRCK ratio shall be a multiple of 2 * CONFIG.SWIDTH, which can be formulated as:

```
Integer = (CONFIG.RATIO / (2 * CONFIG.SWIDTH))
```

The MCK signal can be routed to an output pin (specified in PSEL.MCK) to supply external I^2S devices that require the MCK to be supplied from the outside.

When operating in Slave mode, the I^2S module does not use the MCK and the MCK generator does not need to be enabled.

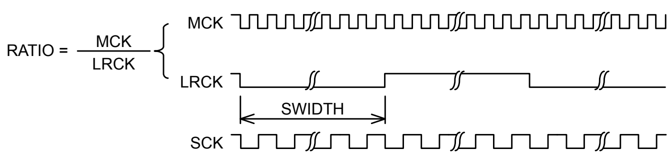

Figure 24: Relation between RATIO, MCK and LRCK.

Desired LRCK [Hz]	CONFIG.SWID'	CONFIG.RATIO	CONFIG.MCKF	MCK [Hz]	LRCK [Hz]	LRCK error [%]
16000	16Bit	32X	32MDIV63	507936.5	15873.0	-0.8
16000	16Bit	64X	32MDIV31	1032258.1	16129.0	0.8
16000	16Bit	256X	32MDIV8	4000000.0	15625.0	-2.3
32000	16Bit	32X	32MDIV31	1032258.1	32258.1	0.8
32000	16Bit	64X	32MDIV16	2000000.0	31250.0	-2.3
44100	16Bit	32X	32MDIV23	1391304.3	43478.3	-1.4
44100	16Bit	64X	32MDIV11	2909090.9	45454.5	3.1

Table 44: Configuration examples

6.7.6 Width, alignment and format

The CONFIG.SWIDTH register primarily defines the sample width of the data written to memory. In master mode, it then also sets the amount of bits per frame. In Slave mode it controls padding/trimming if required. Left, right, transmitted, and received samples always have the same width. The CONFIG.FORMAT register specifies the position of the data frames with respect to the LRCK edges in both Master and Slave modes.

When using I^2S format, the first bit in a half-frame (containing one left or right sample) gets sampled on the second rising edge of the SCK after a LRCK edge. When using Aligned mode, the first bit in a half-frame gets sampled on the first rising edge of SCK following a LRCK edge.

For data being received on SDIN the sample value can be either right or left-aligned inside a half-frame, as specified in CONFIG.ALIGN on page 135. CONFIG.ALIGN on page 135 affects only the decoding of the incoming samples (SDIN), while the outgoing samples (SDOUT) are always left-aligned (or justified).

When using left-alignment, each half-frame starts with the MSB of the sample value (both for data being sent on SDOUT and received on SDIN).

When using right-alignment, each half-frame of data being received on SDIN ends with the LSB of the sample value, while each half-frame of data being sent on SDOUT starts with the MSB of the sample value (same as for left-alignment).

In Master mode, the size of a half-frame (in number of SCK periods) equals the sample width (in number of bits), and in this case the alignment setting does not care as each half-frame in any case will start with the MSB and end with the LSB of the sample value.

In slave mode, however, the sample width does not need to equal the frame size. This means you might have extra or fewer SCK pulses per half-frame than what the sample width specified in CONFIG.SWIDTH requires.

In the case where we use **left-alignment** and the number of SCK pulses per half-frame is **higher** than the sample width, the following will apply:

- For data received on SDIN, all bits after the LSB of the sample value will be discarded.
- For data sent on SDOUT, all bits after the LSB of the sample value will be 0.

In the case where we use **left-alignment** and the number of SCK pulses per frame is **lower** than the sample width, the following will apply:

- Data sent and received on SDOUT and SDIN will be truncated with the LSBs being removed first.

In the case where we use **right-alignment** and the number of SCK pulses per frame is **higher** than the sample width, the following will apply:

- For data received on SDIN, all bits before the MSB of the sample value will be discarded.
- For data sent on SDOUT, all bits after the LSB of the sample value will be 0 (same behavior as for left-alignment).

In the case where we use **right-alignment** and the number of SCK pulses per frame is **lower** than the sample width, the following will apply:

- Data received on SDIN will be sign-extended to "sample width" number of bits before being written to memory.
- Data sent on SDOUT will be truncated with the LSBs being removed first (same behavior as for left-alignment).

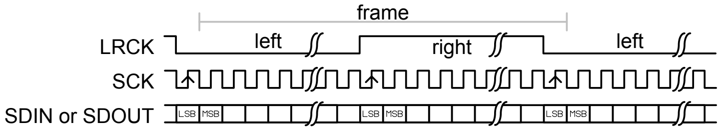

Figure 25: I²S format. CONFIG.SWIDTH equalling half-frame size.

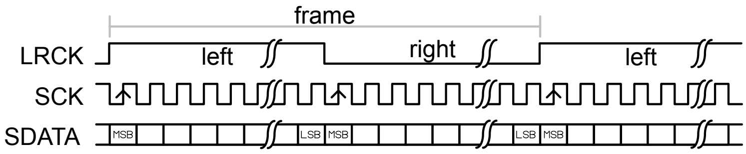

Figure 26: Aligned format. CONFIG.SWIDTH equalling half-frame size.

6.7.7 EasyDMA

The I²S module implements EasyDMA for accessing internal Data RAM without CPU intervention.

The source and destination pointers for the TX and RX data are configured in TXD.PTR on page 136 and RXD.PTR on page 136. The memory pointed to by these pointers will only be read or written when TX or RX are enabled in CONFIG.TXEN on page 134 and CONFIG.RXEN on page 133.

The addresses written to the pointer registers TXD.PTR on page 136 and RXD.PTR on page 136 are double-buffered in hardware, and these double buffers are updated for every RXTXD.MAXCNT on page 137 words (containing one or more samples) read/written from/to memory. The events TXPTRUPD and RXPTRUPD are generated whenever the TXD.PTR and RXD.PTR are transferred to these double buffers.

If TXD.PTR on page 136 is not pointing to the Data RAM region when transmission is enabled, or RXD.PTR on page 136 is not pointing to the Data RAM region when reception is enabled, an EasyDMA transfer may result in a HardFault and/or memory corruption. See Memory on page 20 for more information about the different memory regions.

Due to the nature of I²S, where the number of transmitted samples always equals the number of received samples (at least when both TX and RX are enabled), one common register RXTXD.MAXCNT on page 137 is used for specifying the sizes of these two memory buffers. The size of the buffers is specified in a number of 32-bit words. Such a 32-bit memory word can either contain four 8-bit samples, two 16-bit samples or one right-aligned 24-bit sample sign extended to 32 bit.

In stereo mode (CONFIG.CHANNELS=Stereo), the samples are stored as "left and right sample pairs" in memory. Figure Memory mapping for 8 bit stereo. CONFIG.SWIDTH = 8Bit, CONFIG.CHANNELS = Stereo. on page 124, Memory mapping for 16 bit stereo. CONFIG.SWIDTH = 16Bit, CONFIG.CHANNELS = Stereo. on page 124 and Memory mapping for 24 bit stereo. CONFIG.SWIDTH = 24Bit, CONFIG.CHANNELS = Stereo. on page 125 show how the samples are mapped to memory in this mode. The mapping is valid for both RX and TX.

In mono mode (CONFIG.CHANNELS=Left or Right), RX sample from only one channel in the frame is stored in memory, the other channel sample is ignored. Illustrations Memory mapping for 8 bit mono. CONFIG.SWIDTH = 8Bit, CONFIG.CHANNELS = Left. on page 124, Memory mapping for 16 bit mono, left channel only. CONFIG.SWIDTH = 16Bit, CONFIG.CHANNELS = Left. on page 124 and Memory mapping for 24 bit mono, left channel only. CONFIG.SWIDTH = 24Bit, CONFIG.CHANNELS = Left. on page 125 show how RX samples are mapped to memory in this mode.

For TX, the same outgoing sample read from memory is transmitted on both left and right in a frame, resulting in a mono output stream.

	31–24	23–16	15–8	7–0
x.PTR	Right sample 1	Left sample 1	Right sample 0	Left sample 0
x.PTR + 4	Right sample 3	Left sample 3	Right sample 2	Left sample 2
x.PTR + (n*2) - 4	Right sample n-1	Left sample n-1	Right sample n-2	Left sample n-2

Figure 27: Memory mapping for 8 bit stereo. CONFIG.SWIDTH = 8Bit, CONFIG.CHANNELS = Stereo.

	31–24	23–16	15–8	7–0
x.PTR	Left sample 3	Left sample 2	Left sample 1	Left sample 0
x.PTR + 4	Left sample 7	Left sample 6	Left sample 5	Left sample 4
x.PTR + n - 4	Left sample n-1	Left sample n-2	Left sample n-3	Left sample n-4

Figure 28: Memory mapping for 8 bit mono. CONFIG.SWIDTH = 8Bit, CONFIG.CHANNELS = Left.

	31–16	15–0
x.PTR	Right sample 0	Left sample 0
x.PTR + 4	Right sample 1	Left sample 1
x.PTR + (n*4) - 4	Right sample n - 1	Left sample n - 1

Figure 29: Memory mapping for 16 bit stereo. CONFIG.SWIDTH = 16Bit, CONFIG.CHANNELS = Stereo.

	31–16	15–0
x.PTR	Left sample 1	Left sample 0
x.PTR + 4	Left sample 3	Left sample 2
x.PTR + (n*2) - 4	Left sample n - 1	Left sample n - 2

Figure 30: Memory mapping for 16 bit mono, left channel only. CONFIG.SWIDTH = 16Bit, CONFIG.CHANNELS = Left.

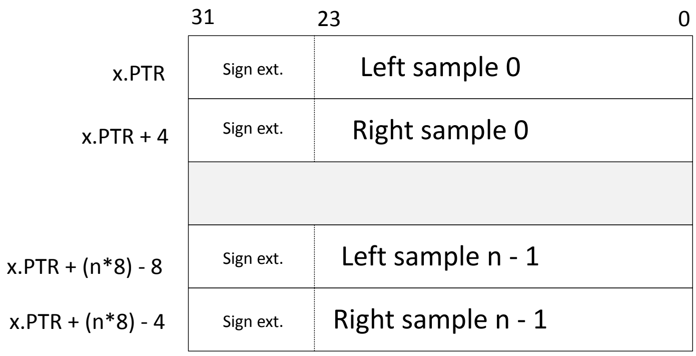

Figure 31: Memory mapping for 24 bit stereo. CONFIG.SWIDTH = 24Bit, CONFIG.CHANNELS = Stereo.

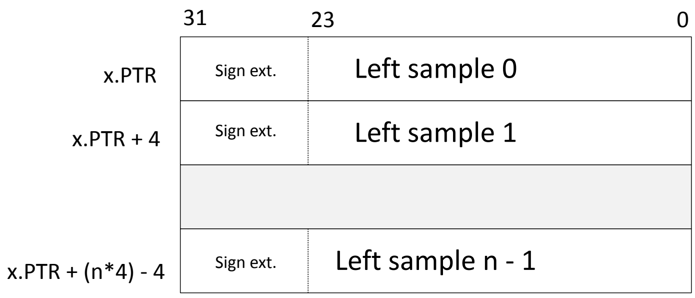

Figure 32: Memory mapping for 24 bit mono, left channel only. CONFIG.SWIDTH = 24Bit, CONFIG.CHANNELS = Left.

6.7.8 Module operation

Described here is a typical operating procedure for the I^2S module.

1. Configure the I²S module using the CONFIG registers

```
// Enable reception
NRF_I2S->CONFIG.RXEN = (I2S_CONFIG_RXEN_RXEN_Enabled <<
                                       I2S_CONFIG_RXEN_RXEN_Pos);
// Enable transmission
NRF_I2S->CONFIG.TXEN = (I2S_CONFIG_TXEN_TXEN_Enabled <<
                                       I2S_CONFIG_TXEN_TXEN_Pos);
// Enable MCK generator
NRF_I2S->CONFIG.MCKEN = (I2S_CONFIG_MCKEN_MCKEN_Enabled <<
                                       I2S_CONFIG_MCKEN_MCKEN_Pos);
// MCKFREQ = 4 MHz
NRF_I2S->CONFIG.MCKFREQ = I2S_CONFIG_MCKFREQ_MCKFREQ_32MDIV8 <<
                                       I2S_CONFIG_MCKFREQ_MCKFREQ_Pos;
// Ratio = 256
NRF_I2S->CONFIG.RATIO = I2S_CONFIG_RATIO_RATIO_256X <<
                                       I2S_CONFIG_RATIO_RATIO_Pos;
// MCKFREQ = 4 MHz and Ratio = 256 gives sample rate = 15.625 ks/s
// Sample width = 16 bit
NRF_I2S->CONFIG.SWIDTH = I2S_CONFIG_SWIDTH_SWIDTH_16Bit <<
                                       I2S_CONFIG_SWIDTH_SWIDTH_Pos;
// Alignment = Left
NRF_I2S->CONFIG.ALIGN = I2S_CONFIG_ALIGN_ALIGN_Left <<
                                       I2S_CONFIG_ALIGN_ALIGN_Pos;
// Format = I2S
NRF_I2S->CONFIG.FORMAT = I2S_CONFIG_FORMAT_FORMAT_I2S <<
                                       I2S_CONFIG_FORMAT_FORMAT_Pos;
// Use stereo
NRF_I2S->CONFIG.CHANNELS = I2S_CONFIG_CHANNELS_CHANNELS_Stereo <<
                                       I2S_CONFIG_CHANNELS_CHANNELS_Pos;
```

2. Map IO pins using the PINSEL registers

```
// MCK routed to pin 0
NRF_I2S->PSEL.MCK = (0 << I2S_PSEL_MCK_PIN_Pos) |
                    (I2S_PSEL_MCK_CONNECT_Connected <<
                                             I2S_PSEL_MCK_CONNECT_Pos);
// SCK routed to pin 1
NRF_I2S->PSEL.SCK = (1 << I2S_PSEL_SCK_PIN_Pos) |
                    (I2S_PSEL_SCK_CONNECT_Connected <<
                                             I2S_PSEL_SCK_CONNECT_Pos);
// LRCK routed to pin 2
NRF_I2S->PSEL.LRCK = (2 << I2S_PSEL_LRCK_PIN_Pos) |
                     (I2S_PSEL_LRCK_CONNECT_Connected <<
                                             I2S_PSEL_LRCK_CONNECT_Pos);
// SDOUT routed to pin 3
NRF_I2S->PSEL.SDOUT = (3 << I2S_PSEL_SDOUT_PIN_Pos) |
                      (I2S_PSEL_SDOUT_CONNECT_Connected <<
                                             I2S_PSEL_SDOUT_CONNECT_Pos);
// SDIN routed on pin 4
NRF_I2S->PSEL.SDIN = (4 << I2S_PSEL_SDIN_PIN_Pos) |
                     (I2S_PSEL_SDIN_CONNECT_Connected <<
                                             I2S_PSEL_SDIN_CONNECT_Pos);
```

3. Configure TX and RX data pointers using the TXD, RXD and RXTXD registers

```
NRF_I2S->TXD.PTR = my_tx_buf;
NRF_I2S->RXD.PTR = my_rx_buf;
NRF_I2S->TXD.MAXCNT = MY_BUF_SIZE;
```

4. Enable the I²S module using the ENABLE register

```
NRF_I2S->ENABLE = 1;
```

5. Start audio streaming using the START task

```
NRF_I2S->TASKS_START = 1;
```

6. Handle received and transmitted data when receiving the TXPTRUPD and RXPTRUPD events

```
if(NRF_I2S->EVENTS_TXPTRUPD  != 0)
{
    NRF_I2S->TXD.PTR = my_next_tx_buf;
    NRF_I2S->EVENTS_TXPTRUPD = 0;
}

if(NRF_I2S->EVENTS_RXPTRUPD != 0)
{
    NRF_I2S->RXD.PTR = my_next_rx_buf;
    NRF_I2S->EVENTS_RXPTRUPD = 0;
}
```

6.7.9 Pin configuration

The MCK, SCK, LRCK, SDIN and SDOUT signals associated with the I²S module are mapped to physical pins according to the pin numbers specified in the PSEL.x registers.

These pins are acquired whenever the I²S module is enabled through the register ENABLE on page 133.

When a pin is acquired by the I²S module, the direction of the pin (input or output) will be configured automatically, and any pin direction setting done in the GPIO module will be overridden. The directions for the various I²S pins are shown below in GPIO configuration before enabling peripheral (master mode) on page 127 and GPIO configuration before enabling peripheral (slave mode) on page 128.

To secure correct signal levels on the pins when the system is in OFF mode, and when the I²S module is disabled, these pins must be configured in the GPIO peripheral directly.

I²S signal	I²S pin	Direction	Output value	Comment
MCK	As specified in PSEL.MCK	Output	0	
LRCK	As specified in PSEL.LRCK	Output	0	
SCK	As specified in PSEL.SCK	Output	0	
SDIN	As specified in PSEL.SDIN	Input	Not applicable	
SDOUT	As specified in PSEL.SDOUT	Output	0	

Table 45: GPIO configuration before enabling peripheral (master mode)

I^2S signal	I^2S pin	Direction	Output value	Comment
MCK	As specified in PSEL.MCK	Output	0	
LRCK	As specified in PSEL.LRCK	Input	Not applicable	
SCK	As specified in PSEL.SCK	Input	Not applicable	
SDIN	As specified in PSEL.SDIN	Input	Not applicable	
SDOUT	As specified in PSEL.SDOUT	Output	0	

Table 46: GPIO configuration before enabling peripheral (slave mode)

6.7.10 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50028000 0x40028000	I2S	I2S : S I2S : NS	US	SA	Inter-IC Sound	

Table 47: Instances

Register	Offset	Security	Description
TASKS_START	0x000		Starts continuous I2S transfer. Also starts MCK generator when this is enabled.
TASKS_STOP	0x004		Stops I2S transfer. Also stops MCK generator. Triggering this task will cause the STOPPED event to be generated.
SUBSCRIBE_START	0x080		Subscribe configuration for task START
SUBSCRIBE_STOP	0x084		Subscribe configuration for task STOP
EVENTS_RXPTRUPD	0x104		The RXD.PTR register has been copied to internal double-buffers. When the I2S module is started and RX is enabled, this event will be generated for every RXTXD.MAXCNT words that are received on the SDIN pin.
EVENTS_STOPPED	0x108		I2S transfer stopped.
EVENTS_TXPTRUPD	0x114		The TDX.PTR register has been copied to internal double-buffers. When the I2S module is started and TX is enabled, this event will be generated for every RXTXD.MAXCNT words that are sent on the SDOUT pin.
PUBLISH_RXPTRUPD	0x184		Publish configuration for event RXPTRUPD
PUBLISH_STOPPED	0x188		Publish configuration for event STOPPED
PUBLISH_TXPTRUPD	0x194		Publish configuration for event TXPTRUPD
INTEN	0x300		Enable or disable interrupt
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
ENABLE	0x500		Enable I2S module.
CONFIG.MODE	0x504		I2S mode.
CONFIG.RXEN	0x508		Reception (RX) enable.
CONFIG.TXEN	0x50C		Transmission (TX) enable.
CONFIG.MCKEN	0x510		Master clock generator enable.
CONFIG.MCKFREQ	0x514		Master clock generator frequency.
CONFIG.RATIO	0x518		MCK / LRCK ratio.
CONFIG.SWIDTH	0x51C		Sample width.
CONFIG.ALIGN	0x520		Alignment of sample within a frame.
CONFIG.FORMAT	0x524		Frame format.
CONFIG.CHANNELS	0x528		Enable channels.
RXD.PTR	0x538		Receive buffer RAM start address.
TXD.PTR	0x540		Transmit buffer RAM start address.
RXTXD.MAXCNT	0x550		Size of RXD and TXD buffers.
PSEL.MCK	0x560		Pin select for MCK signal.
PSEL.SCK	0x564		Pin select for SCK signal.
PSEL.LRCK	0x568		Pin select for LRCK signal.

Register	Offset	Security	Description
PSEL.SDIN	0x56C		Pin select for SDIN signal.
PSEL.SDOUT	0x570		Pin select for SDOUT signal.

Table 48: Register overview

6.7.10.1 TASKS_START

Address offset: 0x000

Starts continuous I2S transfer. Also starts MCK generator when this is enabled.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_START			Starts continuous I2S transfer. Also starts MCK generator when this is enabled.
			Trigger	1	Trigger task

6.7.10.2 TASKS_STOP

Address offset: 0x004

Stops I2S transfer. Also stops MCK generator. Triggering this task will cause the STOPPED event to be generated.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_STOP			Stops I2S transfer. Also stops MCK generator. Triggering this task will cause the STOPPED event to be generated.
			Trigger	1	Trigger task

6.7.10.3 SUBSCRIBE_START

Address offset: 0x080

Subscribe configuration for task START

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task START will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.7.10.4 SUBSCRIBE_STOP

Address offset: 0x084

Subscribe configuration for task STOP

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STOP will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.7.10.5 EVENTS_RXPTRUPD

Address offset: 0x104

The RXD.PTR register has been copied to internal double-buffers. When the I2S module is started and RX is enabled, this event will be generated for every RXTXD.MAXCNT words that are received on the SDIN pin.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_RXPTRUPD			The RXD.PTR register has been copied to internal double-buffers. When the I2S module is started and RX is enabled, this event will be generated for every RXTXD.MAXCNT words that are received on the SDIN pin.
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.7.10.6 EVENTS_STOPPED

Address offset: 0x108

I2S transfer stopped.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_STOPPED			I2S transfer stopped.
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.7.10.7 EVENTS_TXPTRUPD

Address offset: 0x114

The TDX.PTR register has been copied to internal double-buffers. When the I2S module is started and TX is enabled, this event will be generated for every RXTXD.MAXCNT words that are sent on the SDOUT pin.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_TXPTRUPD			The TDX.PTR register has been copied to internal double-buffers. When the I2S module is started and TX is enabled, this event will be generated for every RXTXD.MAXCNT words that are sent on the SDOUT pin.
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.7.10.8 PUBLISH_RXPTRUPD

Address offset: 0x184

Publish configuration for event RXPTRUPD

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event RXPTRUPD will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.7.10.9 PUBLISH_STOPPED

Address offset: 0x188

Publish configuration for event STOPPED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event STOPPED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.7.10.10 PUBLISH_TXPTRUPD

Address offset: 0x194

Publish configuration for event TXPTRUPD

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event TXPTRUPD will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.7.10.11 INTEN

Address offset: 0x300

Enable or disable interrupt

Bit number	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
ID																											F			C	B	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
B	RW	RXPTRUPD			Enable or disable interrupt for event RXPTRUPD
			Disabled	0	Disable
			Enabled	1	Enable
C	RW	STOPPED			Enable or disable interrupt for event STOPPED
			Disabled	0	Disable
			Enabled	1	Enable
F	RW	TXPTRUPD			Enable or disable interrupt for event TXPTRUPD
			Disabled	0	Disable
			Enabled	1	Enable

6.7.10.12 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID																											F			C	B	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
B	RW	RXPTRUPD			Write '1' to enable interrupt for event RXPTRUPD
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	STOPPED			Write '1' to enable interrupt for event STOPPED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
F	RW	TXPTRUPD			Write '1' to enable interrupt for event TXPTRUPD
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.7.10.13 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID																											F			C	B	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
B	RW	RXPTRUPD			Write '1' to disable interrupt for event RXPTRUPD
			Clear	1	Disable

Bit number	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
ID																											F			C	B	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	STOPPED			Write '1' to disable interrupt for event STOPPED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
F	RW	TXPTRUPD			Write '1' to disable interrupt for event TXPTRUPD
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.7.10.14 ENABLE

Address offset: 0x500

Enable I2S module.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ENABLE			Enable I2S module.
			Disabled	0	Disable
			Enabled	1	Enable

6.7.10.15 CONFIG.MODE

Address offset: 0x504

I2S mode.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MODE			I2S mode.
			Master	0	Master mode. SCK and LRCK generated from internal master clcok (MCK) and output on pins defined by PSEL.xxx.
			Slave	1	Slave mode. SCK and LRCK generated by external master and received on pins defined by PSEL.xxx

6.7.10.16 CONFIG.RXEN

Address offset: 0x508

Reception (RX) enable.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	RXEN			Reception (RX) enable.
			Disabled	0	Reception disabled and now data will be written to the RXD.PTR address.
			Enabled	1	Reception enabled.

6.7.10.17 CONFIG.TXEN

Address offset: 0x50C

Transmission (TX) enable.

Bit number	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
ID																																A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	TXEN			Transmission (TX) enable.
			Disabled	0	Transmission disabled and now data will be read from the RXD.TXD address.
			Enabled	1	Transmission enabled.

6.7.10.18 CONFIG.MCKEN

Address offset: 0x510

Master clock generator enable.

Bit number	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
ID																																A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	MCKEN			Master clock generator enable.
			Disabled	0	Master clock generator disabled and PSEL.MCK not connected(available as GPIO).
			Enabled	1	Master clock generator running and MCK output on PSEL.MCK.

6.7.10.19 CONFIG.MCKFREQ

Address offset: 0x514

Master clock generator frequency.

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x20000000	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MCKFREQ			Master clock generator frequency.
			32MDIV8	0x20000000	32 MHz / 8 = 4.0 MHz
			32MDIV10	0x18000000	32 MHz / 10 = 3.2 MHz
			32MDIV11	0x16000000	32 MHz / 11 = 2.9090909 MHz
			32MDIV15	0x11000000	32 MHz / 15 = 2.1333333 MHz

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x20000000	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			32MDIV16	0x10000000	32 MHz / 16 = 2.0 MHz
			32MDIV21	0x0C000000	32 MHz / 21 = 1.5238095
			32MDIV23	0x0B000000	32 MHz / 23 = 1.3913043 MHz
			32MDIV30	0x08800000	32 MHz / 30 = 1.0666667 MHz
			32MDIV31	0x08400000	32 MHz / 31 = 1.0322581 MHz
			32MDIV32	0x08000000	32 MHz / 32 = 1.0 MHz
			32MDIV42	0x06000000	32 MHz / 42 = 0.7619048 MHz
			32MDIV63	0x04100000	32 MHz / 63 = 0.5079365 MHz
			32MDIV125	0x020C0000	32 MHz / 125 = 0.256 MHz

6.7.10.20 CONFIG.RATIO

Address offset: 0x518

MCK / LRCK ratio.

Bit number	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
ID																													A	A	A	A
Reset 0x00000006	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	RATIO			MCK / LRCK ratio.
			32X	0	LRCK = MCK / 32
			48X	1	LRCK = MCK / 48
			64X	2	LRCK = MCK / 64
			96X	3	LRCK = MCK / 96
			128X	4	LRCK = MCK / 128
			192X	5	LRCK = MCK / 192
			256X	6	LRCK = MCK / 256
			384X	7	LRCK = MCK / 384
			512X	8	LRCK = MCK / 512

6.7.10.21 CONFIG.SWIDTH

Address offset: 0x51C

Sample width.

Bit number	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
ID																															A	A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	SWIDTH			Sample width.
			8Bit	0	8 bit.
			16Bit	1	16 bit.
			24Bit	2	24 bit.

6.7.10.22 CONFIG.ALIGN

Address offset: 0x520

Alignment of sample within a frame.

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	ALIGN			Alignment of sample within a frame.
			Left	0	Left-aligned.
			Right	1	Right-aligned.

6.7.10.23 CONFIG.FORMAT

Address offset: 0x524

Frame format.

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	FORMAT			Frame format.
			I2S	0	Original I2S format.
			Aligned	1	Alternate (left- or right-aligned) format.

6.7.10.24 CONFIG.CHANNELS

Address offset: 0x528

Enable channels.

Bit number	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
ID																															A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHANNELS			Enable channels.
			Stereo	0	Stereo.
			Left	1	Left only.
			Right	2	Right only.

6.7.10.25 RXD.PTR

Address offset: 0x538

Receive buffer RAM start address.

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			Receive buffer Data RAM start address. When receiving, words containing samples will be written to this address. This address is a word aligned Data RAM address.

6.7.10.26 TXD.PTR

Address offset: 0x540

Transmit buffer RAM start address.

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			Transmit buffer Data RAM start address. When transmitting, words containing samples will be fetched from this address. This address is a word aligned Data RAM address.

6.7.10.27 RXTXD.MAXCNT

Address offset: 0x550

Size of RXD and TXD buffers.

Bit number	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
ID																			A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXCNT			Size of RXD and TXD buffers in number of 32 bit words.

6.7.10.28 PSEL.MCK

Address offset: 0x560

Pin select for MCK signal.

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.7.10.29 PSEL.SCK

Address offset: 0x564

Pin select for SCK signal.

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.7.10.30 PSEL.LRCK

Address offset: 0x568

Pin select for LRCK signal.

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.7.10.31 PSEL.SDIN

Address offset: 0x56C

Pin select for SDIN signal.

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.7.10.32 PSEL.SDOUT

Address offset: 0x570

Pin select for SDOUT signal.

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.7.11 Electrical specification

6.7.11.1 I2S timing specification

Symbol	Description	Min.	Typ.	Max.	Units
t_{S_SDIN}	SDIN setup time before SCK rising	20			ns
t_{H_SDIN}	SDIN hold time after SCK rising	15			ns
t_{S_SDOUT}	SDOUT setup time after SCK falling	40			ns
t_{H_SDOUT}	SDOUT hold time before SCK falling	6			ns
t_{SCK_LRCK}	SCLK falling to LRCK edge	-5	0	5	ns
f_{MCK}	MCK frequency			4000	kHz
f_{LRCK}	LRCK frequency			48	kHz
f_{SCK}	SCK frequency			2000	kHz
DC_{CK}	Clock duty cycle (MCK, LRCK, SCK)	45		55	%

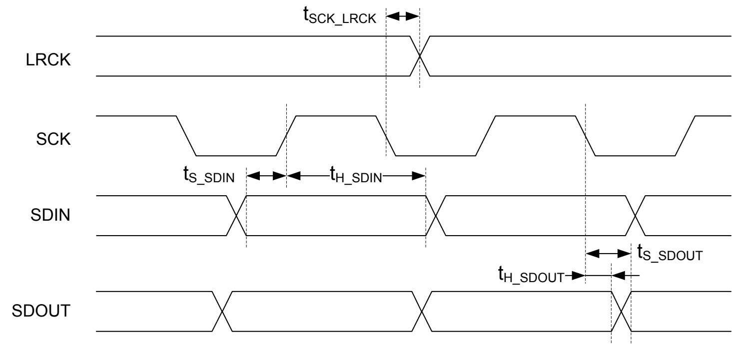

Figure 33: I2S timing diagram

6.8 KMU — Key management unit

The key management unit (KMU) enforces access policies to a subset region of user information configuration register (UICR). This subset region is used for storing cryptographic key values inside the key slots, which the CPU has no access to.

In total there are 128 key slots available, where each key slot can store one 128-bit key value together with an access policy and a destination address for the key value. Multiple key slots can be combined in order to support key sizes larger than 128 bits. The access policy of a key slot governs if and how a key value can be used, while the destination address determines where in the memory map the KMU pushes the key value upon a request from the CPU.

Key slots can be configured to be pushed directly into write-only key registers in cryptographic accelerators, like e.g. CryptoCell, without exposing the key value itself to the CPU. This enables the CPU to use the key values stored inside the key slots for cryptographic operations without being exposed to the key value.

Access to the KMU, and the key slots in the UICR, is only allowed from secure mode.

6.8.1 Functional view

From a functional view the UICR is divided into two different regions, one-time programmable (OTP) memory and key storage.

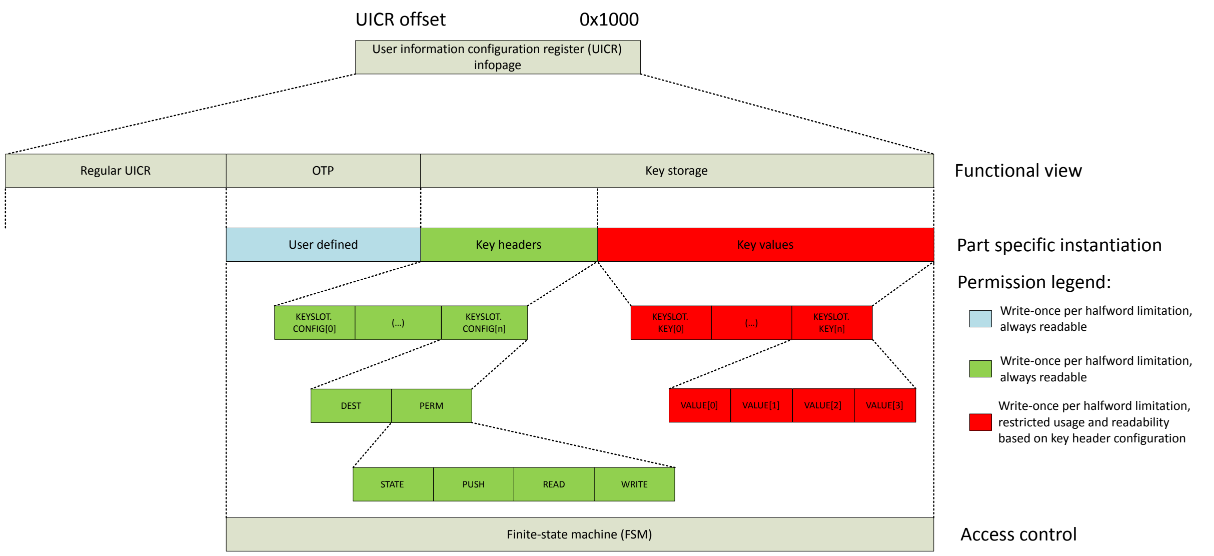

Figure 34: Memory map overview

OTP

One-time programmable (OTP) memory is typically used for holding values that are written once, and then never to be changed again throughout the product lifetime. The OTP region of UICR is emulated by placing a write-once per halfword limitation on registers defined here.

Key storage

The key storage region contains multiple key slots, where each slot consists of a key header and an associated key value. The key value is limited to 128 bits. Any key size greater than 128 bits must be divided and distributed over multiple key slot instances.

Key headers are allocated an address range of 0x400 in the UICR memory map, allowing a total of 128 keys to be addressable inside the key storage region.

> **Note:** The use of the key storage region in UICR should be limited to keys with a certain life span, and not per-session derived keys where the CPU is involved in the key exchange.

6.8.2 Access control

Access control to the underlying UICR infopage in flash is enforced by a hardware finite-state machine (FSM). The FSM can allow or block transactions, depending both on the security of the transaction (secure or non-secure) and on the type of register being written and/or read.

Access type	Key headers	Key values
Read	Allowed	Restricted
Write	Restricted	Restricted

Table 49: Access control

Any restricted access requires an explicit key slot selection through the KMU register interface. Any illegal access to restricted key slot registers will be blocked and word `0xDEADDEAD` will be returned on the AHB.

The OTP region has individual access control behavior, while access control to the key storage region is configured on a per key slot basis. The KMU FSM operates on only one key slot instance at a time, and the permissions and the usage restriction for a key value associated with a key slot can be configured individually.

> **Note:** Even if the KMU can be configured as non-secure, all non-secure transactions will be blocked.

6.8.3 Protecting the UICR content

The UICR content can be protected against device-internal NVMC.ERASEALL requests, in addition to device-external ERASEALL requests, through the CTRL-AP interface. This feature is useful if the firmware designers want to prevent the OTP region from being erased.

Since enabling this step will permanently disable erase for the UICR, the procedure requires an implementation defined 32-bit word to be written into the UICR's ERASEPROTECT register.

In case of a field return handling, it is still possible to erase the UICR even if the ERASEPROTECT is set. If this functionality is desired, the secure boot code must implement a secure communication channel over the CTRL-AP mailbox interface. Upon successful authentication of the external party, the secure boot code can temporarily re-enable the CTRL-AP ERASEALL functionality.

6.8.4 Usage

This section describes the specific KMU and UICR behavior in more detail, to help the reader get a better overview of KMU's features and the intended usage.

6.8.4.1 OTP

The OTP region of the UICR contains a user-defined static configuration of the device. The KMU emulates the OTP functionality by placing a write-once per halfword limitation of registers defined in this region, i.e. only halfwords containing all '1's can be written.

An OTP write transaction must consist of a full 32-bit word. Both halfwords can either be written simultaneously or one at a time. The KMU FSM will block any write to a halfword in the OTP region, if the initial value of this halfword is not 0xFFFF. When writing halfwords one at a time, the non-active halfword must be masked as 0xFFFF, otherwise the request will be blocked. For example, writing 0x1234XXXX to an OTP destination address which already contains the value 0xFFFFAABB, must be configured as 0x1234FFFF. The OTP destination address will contain the value 0x1234AABB after both write transactions have been processed.

The KMU will also only allow secure AHB write transactions into the OTP region of the UICR. Any AHB write transaction to this region that does not satisfy the above requirements will be ignored, and the STATUS.BLOCKED register will be set to '1'.

6.8.4.2 Key storage

The key storage region of the UICR can contain multiple keys of different type, including symmetrical keys, hashes, public/private key pairs and other device secrets. One of the key features of the KMU, is that these device secrets can be installed and made available for use in cryptographic operations without revealing the actual secret values.

Keys in this region will typically have a certain life span. The region is not designed to be used for per-session derived keys where the non-secure side (i.e. application) is participating in the key exchange.

All key storage is done through the concept of multiple key slots, where each key slot instance consists of one key header and an associated key value. Each key header supports the configuration of usage permissions and an optional secure destination address.

The key header secure destination address option enables the KMU to push the associated key value over a dedicated secure APB to a pre-configured secure location within the memory map. Such locations typically include a write-only key register of the hardware cryptograhic accelerator, allowing the KMU to distribute keys within the system without compromising the key values.

One key slot instance can store a key value of maximum 128 bits. If a key size exceeds this limit, the key value itself must be split over multiple key slot instances.

The following usage and read permissions scheme is applicable for each key slot:

State	Push	Read	Write	Description
Active (1)	Enabled (1)	Enabled (1)	Enabled (1)	Default flash erase value. Key slot cannot be pushed, write is enabled.
Active (1)	Enabled (1)	Enabled (1)	Disabled (0)	Key slot is active, push is enabled. Key slot VALUE registers can be read, but write is disabled.
Active (1)	Enabled (1)	Disabled (0)	Disabled (0)	Key slot is active, push is enabled. Read and write to key slot VALUE registers are disabled.
Active (1)	Disabled (0)	Enabled (1)	Disabled (0)	Key slot is active, push is disabled. Key slot VALUE registers can be read, but write is disabled.
Revoked (0)	-	-	-	Key slot is revoked. Cannot be read or pushed over secure APB regardless of the permission settings.

Table 50: Valid key slot permission schemes

6.8.4.2.1 Selecting a key slot

The KMU FSM is designed to process only one key slot at a time, effectively operating as a memory protection unit for the key storage region. Whenever a key slot is selected, the KMU will allow access to writing, reading, and/or pushing the associated key value according to the selected slot configuration.

A key slot must be selected prior to use, by writing the key slot ID into the KMU SELECTKEYSLOT register. Because the reset value of this register is 0x00000000, there is no key slot associated with ID=0 and no slot is selected by default. All key slots are addressed using IDs from 1 to 128.

SELECTED status is set when a key slot is selected, and a read or write acccess to that keyslot occurs.

BLOCKED status is set when any illegal access to key slot registers is detected.

When the use of the particular key slot is stopped, the key slot selection in SELECTKEYSLOT must be set back to '0'.

By default, all KMU key slots will consist of a 128-bit key value of '1's, where the key headers have no secure destination address, or any usage and read restrictions.

6.8.4.2.2 Writing to a key slot

Writing a key slot into UICR is a five-step process.

1. Select which key slot the KMU shall operate on by writing the desired key slot ID into KMU->SELECTKEYSLOT. The selected key slot must be empty in order to add a new entry to UICR.
2. If the key value shall be pushable over secure APB, the destination address of the recipient must be configured in register KEYSLOT.CONFIG[ID-1].DEST.
3. Write the 128-bit key value into KEYSLOT.KEY[ID-1].VALUE[0-3].
4. Write the desired key slot permissions into KEYSLOT.CONFIG[ID-1].PERM, including any applicable usage restrictions.
5. Select key slot 0.

In case the total key size is greater than 128 bits, the key value itself must be split into 128-bit segments and written to multiple key slot instances. Steps 1 through 5 above must be repeated for the entire key size.

> **Note:** If a key slot is configured as readable, and KEYSLOT.CONFIG[ID-1].DEST is not to be used, it is recommended to disable the push bit in KEYSLOT.CONFIG[ID-1].PERM when configuring key slot permissions.

> **Note:** A key value distributed over multiple key slots should use the same key slot configuration in its key headers, but the secure destination address for each key slot instance must be incremented by 4 words (128 bits) for each key slot instance spanned.

Note: Write to flash must be enabled in NVMC->CONFIG prior to writing keys to flash, and subsequently disabled once writing is complete.

Steps 1 through 5 above will be blocked if any of the following violations are detected:

- No key slot selected
- Non-empty key slot selected
- NVM destination address not empty
- AHB write to KEYSLOT.KEY[ID-1].VALUE[0-3] registers not belonging to selected key slot

6.8.4.2.3 Reading a key value

Key slots that are configured as readable can have their key value read directly from the UICR memory map by the CPU.

Readable keys are typically used during the secure boot sequence, where the CPU is involved in falsifying or verifying the integrity of the system. Since the CPU is involved in this decision process, it makes little sense not to trust the CPU having access to the actual key value but ultimately trust the decision of the integrity check. Another use-case for readable keys is if the key type in question does not have a HW peripheral in the platform that is able to accept such keys over secure APB.

Reading a key value from the UICR is a three-step process:

1. Select the key slot which the KMU shall operate on by writing the desired key slot ID into KMU->SELECTKEYSLOT.
2. If STATE and READ permission requirements are fulfilled as defined in KEYSLOT.CONFIG[ID-1].PERM, the key value can be read from region KEYSLOT.KEY[ID-1].VALUE[0-3] for selected key slot.
3. Select key slot 0.

Step 2 will be blocked and word 0xDEADDEAD will be returned on AHB if any of the following violations are detected:

- No key slot selected
- Key slot not configured as readable
- Key slot is revoked
- AHB read to KEYSLOT.KEY[ID-1].VALUE[0-3] registers not belonging to selected key slot

6.8.4.2.4 Push over secure APB

Key slots that are configured as non-readable cannot be read by the CPU regardless of the mode the system is in, and must be pushed over secure APB in order to use the key value for cryptographic operations.

The secure APB destination address is set in the key slot configuration DEST register. Such destination addresses are typically write-only key registers in a hardware cryptographic accelerators memory map. The secure APB allows key slots to be utilized by the software side, without exposing the key value itself.

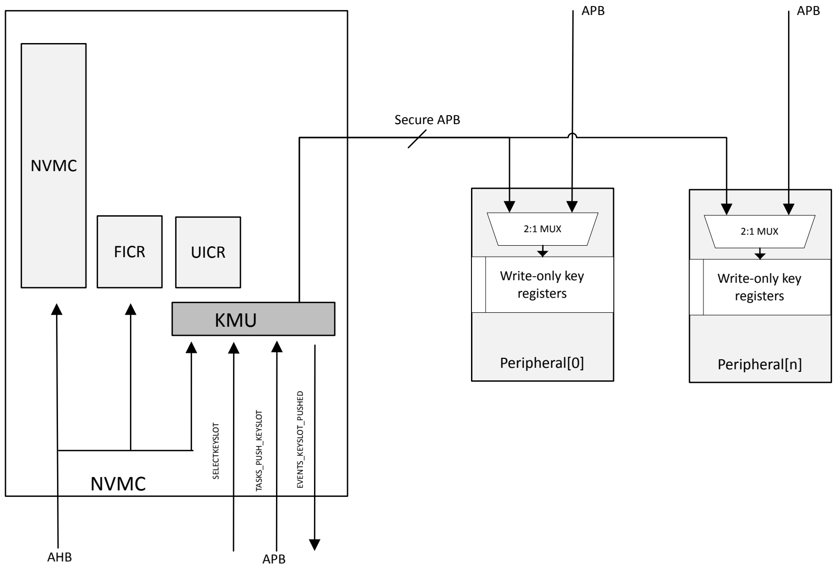

Figure 35: Tasks and events pattern for key slots

Pushing a key slot over secure APB is a four-step process:

1. Select the key slot on which the KMU shall operate by writing the desired key slot ID into KMU->SELECTKEYSLOT.
2. Start TASKS_PUSH_KEYSLOT to initiate a secure APB transaction, writing the 128-bit key value associated with the selected key slot into address defined in KEYSLOT.CONFIG[ID-1].DEST.
3. After completing the secure APB transaction, the 128-bit key value is ready for use by the peripheral and EVENTS_KEYSLOT_PUSHED is triggered.
4. Select key slot 0.

> **Note:** If a key value is distributed over multiple key slots due to its key size, exceeding the maximum 128-bit key value limitation, then each distributed key slot must be pushed individually in order to transfer the entire key value over secure APB.

Step 3 will trigger other events than EVENTS_KEYSLOT_PUSHED if the following violations are detected:

- EVENTS_KEYSLOT_ERROR:
  - If no key slot is selected
  - If a key slot has no destination address configured
  - If when pushing a key slot, flash or peripheral returns an error
  - If pushing a key slot when push permissions are disabled
  - If attempting to push a key slot with default permissions
- EVENTS_KEYSLOT_REVOKED if a key slot is marked as revoked in its key header configuration

6.8.4.2.5 Revoking the key slots

All key slots within the key storage area can be marked as revoked.

To revoke any key slots, write to the STATE field in the KEYSLOT.CONFIG[ID-1].PERM register. The following rules apply to keys that have been revoked:

1. Key slots that have the PUSH field enabled in PERM register can no longer be pushed. If a revoked key slot is selected and task TASKS_PUSH_KEYSLOT is started, the event EVENTS_KEYSLOT_REVOKED is triggered.

2. Key slots that have the READ field enabled in PERM register can no longer be read. Any read operation to a revoked key value will return word 0xDEADDEAD.
3. Previously pushed key values stored in a peripheral write-only key register are not affected by key revocation. If secure code wants to enforce that a revoked key is no longer usable by a peripheral for cryptographic operations, the secure code should disable or reset the peripheral in question.

6.8.4.3 STATUS register

The KMU uses a STATUS register to indicate its status of operation. The SELECTED bit will be asserted whenever the currently selected key slot is successfully read from or written to.

All read or write operations to other key slots than what is currently selected in KMU->SELECTKEYSLOT will assert the BLOCKED bit. The BLOCKED bit will also be asserted if the KMU fails to select a key slot, or if a request has been blocked due to an access violation. Normal operation using the KMU should never trigger the BLOCKED bit. If this bit is triggered during the development phase, it indicates that the code is using the KMU incorrectly.

The STATUS register is reset every time register SELECTKEYSLOT is written.

6.8.5 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50039000 0x40039000	KMU	KMU : S KMU : NS	SPLIT	NA	Key management unit	

Table 51: Instances

Register	Offset	Security	Description
TASKS_PUSH_KEYSLOT	0x0000		Push a key slot over secure APB
EVENTS_KEYSLOT_PUSHED	0x100		Key slot successfully pushed over secure APB
EVENTS_KEYSLOT_REVOKED	0x104		Key slot has been revoked and cannot be tasked for selection
EVENTS_KEYSLOT_ERROR	0x108		No key slot selected, no destination address defined, or error during push operation
INTEN	0x300		Enable or disable interrupt
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
INTPEND	0x30C		Pending interrupts
STATUS	0x40C		Status bits for KMU operation
SELECTKEYSLOT	0x500		Select key slot to be read over AHB or pushed over secure APB when TASKS_PUSH_KEYSLOT is started

Table 52: Register overview

6.8.5.1 TASKS_PUSH_KEYSLOT

Address offset: 0x0000

Push a key slot over secure APB

Bit number				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
ID				A
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_PUSH_KEYSLOT			Push a key slot over secure APB
			Trigger	1	Trigger task

6.8.5.2 EVENTS_KEYSLOT_PUSHED

Address offset: 0x100

Key slot successfully pushed over secure APB

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_KEYSLOT_PUSHED			Key slot successfully pushed over secure APB
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.8.5.3 EVENTS_KEYSLOT_REVOKED

Address offset: 0x104

Key slot has been revoked and cannot be tasked for selection

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_KEYSLOT_REVOKED			Key slot has been revoked and cannot be tasked for selection
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.8.5.4 EVENTS_KEYSLOT_ERROR

Address offset: 0x108

No key slot selected, no destination address defined, or error during push operation

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_KEYSLOT_ERROR			No key slot selected, no destination address defined, or error during push operation
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.8.5.5 INTEN

Address offset: 0x300

Enable or disable interrupt

Bit number	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
ID																														C	B	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	KEYSLOT_PUSHED			Enable or disable interrupt for event KEYSLOT_PUSHED

Bit number	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
ID	C B A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
			Disabled	0	Disable
			Enabled	1	Enable
B	RW	KEYSLOT_REVOKED			Enable or disable interrupt for event KEYSLOT_REVOKED
			Disabled	0	Disable
			Enabled	1	Enable
C	RW	KEYSLOT_ERROR			Enable or disable interrupt for event KEYSLOT_ERROR
			Disabled	0	Disable
			Enabled	1	Enable

6.8.5.6 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID	C B A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	KEYSLOT_PUSHED			Write '1' to enable interrupt for event KEYSLOT_PUSHED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	KEYSLOT_REVOKED			Write '1' to enable interrupt for event KEYSLOT_REVOKED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	KEYSLOT_ERROR			Write '1' to enable interrupt for event KEYSLOT_ERROR
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.8.5.7 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID	C B A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	KEYSLOT_PUSHED			Write '1' to disable interrupt for event KEYSLOT_PUSHED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	KEYSLOT_REVOKED			Write '1' to disable interrupt for event KEYSLOT_REVOKED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	KEYSLOT_ERROR			Write '1' to disable interrupt for event KEYSLOT_ERROR

Bit number	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
ID																														C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.8.5.8 INTPEND

Address offset: 0x30C

Pending interrupts

Bit number	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
ID																														C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	KEYSLOT_PUSHED			Read pending status of interrupt for event KEYSLOT_PUSHED
			NotPending	0	Read: Not pending
			Pending	1	Read: Pending
B	R	KEYSLOT_REVOKED			Read pending status of interrupt for event KEYSLOT_REVOKED
			NotPending	0	Read: Not pending
			Pending	1	Read: Pending
C	R	KEYSLOT_ERROR			Read pending status of interrupt for event KEYSLOT_ERROR
			NotPending	0	Read: Not pending
			Pending	1	Read: Pending

6.8.5.9 STATUS

Address offset: 0x40C

Status bits for KMU operation

This register is reset and re-written by the KMU whenever SELECTKEYSLOT is written

Bit number	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
ID																															B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	SELECTED			Key slot ID successfully selected by the KMU
			Disabled	0	No key slot ID selected by KMU
			Enabled	1	Key slot ID successfully selected by KMU
B	R	BLOCKED			Violation status
			Disabled	0	No access violation detected
			Enabled	1	Access violation detected and blocked

6.8.5.10 SELECTKEYSLOT

Address offset: 0x500

Select key slot to be read over AHB or pushed over secure APB when TASKS_PUSH_KEYSLOT is started

Bit number	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
ID																									A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ID			Select key slot ID to be read over AHB, or pushed over secure APB, when TASKS_PUSH_KEYSLOT is started. NOTE: ID=0 is not a valid key slot ID. The 0 ID should be used when the KMU is idle or not in use. NOTE: Index N in UICR->KEYSLOT.KEY[N] and UICR->KEYSLOT.CONFIG[N] corresponds to KMU key slot ID=N+1.

6.9 PDM — Pulse density modulation interface

The pulse density modulation (PDM) module enables input of pulse density modulated signals from external audio frontends, for example, digital microphones. The PDM module generates the PDM clock and supports single-channel or dual-channel (Left and Right) data input. Data is transferred directly to RAM buffers using EasyDMA.

Listed here are the main features for PDM:

- Up to two PDM microphones configured as a Left/Right pair using the same data input
- 16 kHz output sample rate, 16-bit samples
- EasyDMA support for sample buffering
- HW decimation filters
- Selectable ratio of 64 or 80 between PDM_CLK and output sample rate

The PDM module illustrated in PDM module on page 149 is interfacing up to two digital microphones with the PDM interface. It implements EasyDMA, which relieves real-time requirements associated with controlling the PDM slave from a low priority CPU execution context. It also includes all the necessary digital filter elements to produce PCM samples. The PDM module allows continuous audio streaming.

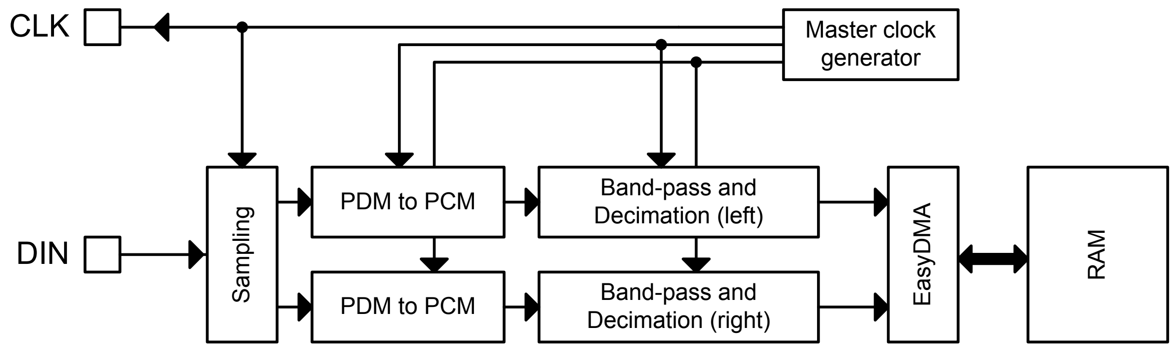

Figure 36: PDM module

6.9.1 Master clock generator

The FREQ field in the master clock's PDMCLKCTRL register allows adjusting the PDM clock's frequency.

The master clock generator does not add any jitter to the HFCLK source chosen. It is recommended (but not mandatory) to use the Xtal as HFCLK source.

6.9.2 Module operation

By default, bits from the left PDM microphone are sampled on PDM_CLK falling edge, bits for the right are sampled on the rising edge of PDM_CLK, resulting in two bitstreams. Each bitstream is fed into a digital filter which converts the PDM stream into 16-bit PCM samples, and filters and down-samples them to reach the appropriate sample rate.

The EDGE field in the MODE register allows swapping Left and Right, so that Left will be sampled on rising edge, and Right on falling.

The PDM module uses EasyDMA to store the samples coming out from the filters into one buffer in RAM.

Depending on the mode chosen in the OPERATION field in the MODE register, memory either contains alternating left and right 16-bit samples (Stereo), or only left 16-bit samples (Mono).

To ensure continuous PDM sampling, it is up to the application to update the EasyDMA destination address pointer as the previous buffer is filled.

The continuous transfer can be started or stopped by sending the START and STOP tasks. STOP becomes effective after the current frame has finished transferring, which will generate the STOPPED event. The STOPPED event indicates that all activity in the module are finished, and that the data is available in RAM (EasyDMA has finished transferring as well). Attempting to restart before receiving the STOPPED event may result in unpredictable behaviour.

6.9.3 Decimation filter

In order to convert the incoming data stream into PCM audio samples, a decimation filter is included in the PDM interface module.

The input of the filter is the two-channel PDM serial stream (with left channel on clock high, right channel on clock low). Depending on the RATIO selected, its output is 2 × 16-bit PCM samples at a sample rate either 64 times or 80 times (depending on the RATIO register) lower than the PDM clock rate.

The filter stage of each channel is followed by a digital volume control, to attenuate or amplify the output samples in a range of -20 dB to +20 dB around the default (reset) setting, defined by $G_{PDM,default}$. The gain is controlled by the GAINL and GAINR registers.

As an example, if the goal is to achieve 2500 RMS output samples (16 bit) with a 1 kHz 90 dBA signal into a -26 dBFS sensitivity PDM microphone, the user will have to sum the PDM module's default gain ($G_{PDM,default}$) and the gain introduced by the microphone and acoustic path of his implementation (an attenuation would translate into a negative gain), and adjust GAINL and GAINR by this amount. Assuming that only the PDM module influences the gain, GAINL and GAINR must be set to $-G_{PDM,default}$ dB to achieve the requirement.

With $G_{PDM,default}$=3.2 dB, and as GAINL and GAINR are expressed in 0.5 dB steps, the closest value to program would be 3.0 dB, which can be calculated as:

```
GAINL = GAINR = (DefaultGain - (2 * 3))
```

Remember to check that the resulting values programmed into GAINL and GAINR fall within MinGain and MaxGain.

6.9.4 EasyDMA

Samples will be written directly to RAM, and EasyDMA must be configured accordingly.

The address pointer for the EasyDMA channel is set in SAMPLE.PTR register. If the destination address set in SAMPLE.PTR is not pointing to the Data RAM region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 20 for more information about the different memory regions.

DMA supports Stereo (Left+Right 16-bit samples) and Mono (Left only) data transfer, depending on setting in the OPERATION field in the MODE register. The samples are stored little endian.

MODE.OPERATION	Bits per sample	Result stored per RAM word	Physical RAM allocated (32 bit words)	Result boundary indexes in RAM	Note
Stereo	32 (2x16)	L+R	ceil(SAMPLE.MAXCNT/2)	R0=[31:16]; L0=[15:0]	Default
Mono	16	2xL	ceil(SAMPLE.MAXCNT/2)	L1=[31:16]; L0=[15:0]	

Table 53: DMA sample storage

The destination buffer in RAM consists of one block, the size of which is set in SAMPLE.MAXCNT register. Format is number of 16-bit samples. The physical RAM allocated is always:

```
(RAM allocation, in bytes) = SAMPLE.MAXCNT * 2;
```

(but the mapping of the samples depends on MODE.OPERATION.

If OPERATION=Stereo, RAM will contain a succession of Left and Right samples.

If OPERATION=Mono, RAM will contain a succession of mono samples.

For a given value of SAMPLE.MAXCNT, the buffer in RAM can contain half the stereo sampling time as compared to the mono sampling time.

The PDM acquisition can be started by the START task, after the SAMPLE.PTR and SAMPLE.MAXCNT registers have been written. When starting the module, it will take some time for the filters to start outputting valid data. Transients from the PDM microphone itself may also occur. The first few samples (typically around 50) might hence contain invalid values or transients. It is therefore advised to discard the first few samples after a PDM start.

As soon as the STARTED event is received, the firmware can write the next SAMPLE.PTR value (this register is double-buffered), to ensure continuous operation.

When the buffer in RAM is filled with samples, an END event is triggered. The firmware can start processing the data in the buffer. Meanwhile, the PDM module starts acquiring data into the new buffer pointed to by SAMPLE.PTR, and sends a new STARTED event, so that the firmware can update SAMPLE.PTR to the next buffer address.

6.9.5 Hardware example

Connect the microphone clock to CLK, and data to DIN.

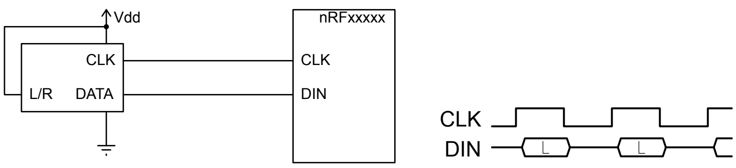

Figure 37: Example of a single PDM microphone, wired as left

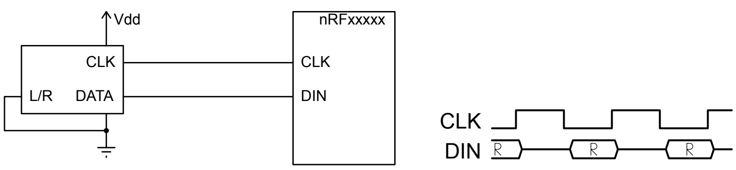

Figure 38: Example of a single PDM microphone, wired as right

Note that in a single-microphone (mono) configuration, depending on the microphone's implementation, either the left or the right channel (sampled at falling or rising CLK edge respectively) will contain reliable data. If two microphones are used, one of them has to be set as left, the other as right (L/R pin tied high or to GND on the respective microphone). It is strongly recommended to use two microphones of exactly the same brand and type so that their timings in left and right operation match.

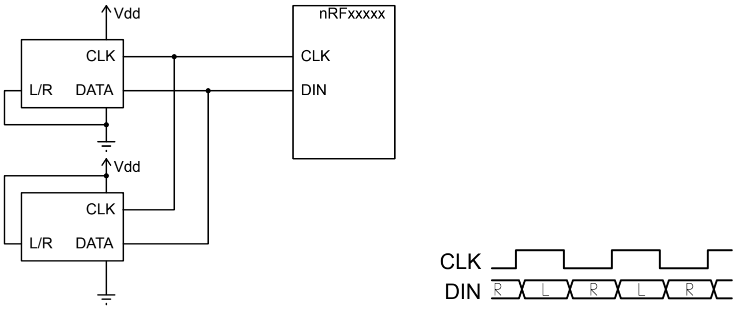

Figure 39: Example of two PDM microphones

6.9.6 Pin configuration

The CLK and DIN signals associated to the PDM module are mapped to physical pins according to the configuration specified in the PSEL.CLK and PSEL.DIN registers respectively. If the CONNECT field in any PSEL register is set to Disconnected, the associated PDM module signal will not be connected to the required physical pins, and will not operate properly.

The PSEL.CLK and PSEL.DIN registers and their configurations are only used as long as the PDM module is enabled, and retained only as long as the device is in System ON mode. See POWER — Power control on page 59 for more information about power modes. When the peripheral is disabled, the pins will behave as regular GPIOs, and use the configuration in their respective OUT bit field and PIN_CNF[n] register.

To ensure correct behaviour in the PDM module, the pins used by the PDM module must be configured in the GPIO peripheral as described in GPIO configuration before enabling peripheral on page 152 before enabling the PDM module. This is to ensure that the pins used by the PDM module are driven correctly if the PDM module itself is temporarily disabled or the device temporarily enters System OFF. This configuration must be retained in the GPIO for the selected I/Os as long as the PDM module is supposed to be connected to an external PDM circuit.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behaviour.

PDM signal	PDM pin	Direction	Output value	Comment
CLK	As specified in PSEL.CLK	Output	0	
DIN	As specified in PSEL.DIN	Input	Not applicable	

Table 54: GPIO configuration before enabling peripheral

6.9.7 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50026000 0x40026000	PDM	PDM : S PDM : NS	US	SA	Pulse density modulation (digital microphone) interface	

Table 55: Instances

Register	Offset	Security	Description
TASKS_START	0x000		Starts continuous PDM transfer
TASKS_STOP	0x004		Stops PDM transfer
SUBSCRIBE_START	0x080		Subscribe configuration for task START
SUBSCRIBE_STOP	0x084		Subscribe configuration for task STOP

Register	Offset	Security	Description
EVENTS_STARTED	0x100		PDM transfer has started
EVENTS_STOPPED	0x104		PDM transfer has finished
EVENTS_END	0x108		The PDM has written the last sample specified by SAMPLE.MAXCNT (or the last sample after a STOP task has been received) to Data RAM
PUBLISH_STARTED	0x180		Publish configuration for event STARTED
PUBLISH_STOPPED	0x184		Publish configuration for event STOPPED
PUBLISH_END	0x188		Publish configuration for event END
INTEN	0x300		Enable or disable interrupt
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
ENABLE	0x500		PDM module enable register
PDMCLKCTRL	0x504		PDM clock generator control
MODE	0x508		Defines the routing of the connected PDM microphones' signals
GAINL	0x518		Left output gain adjustment
GAINR	0x51C		Right output gain adjustment
RATIO	0x520		Selects the ratio between PDM_CLK and output sample rate. Change PDMCLKCTRL accordingly.
PSEL.CLK	0x540		Pin number configuration for PDM CLK signal
PSEL.DIN	0x544		Pin number configuration for PDM DIN signal
SAMPLE.PTR	0x560		RAM address pointer to write samples to with EasyDMA
SAMPLE.MAXCNT	0x564		Number of samples to allocate memory for in EasyDMA mode

Table 56: Register overview

6.9.7.1 TASKS_START

Address offset: 0x000

Starts continuous PDM transfer

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_START			Starts continuous PDM transfer
			Trigger	1	Trigger task

6.9.7.2 TASKS_STOP

Address offset: 0x004

Stops PDM transfer

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_STOP			Stops PDM transfer
			Trigger	1	Trigger task

6.9.7.3 SUBSCRIBE_START

Address offset: 0x080

Subscribe configuration for task START

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task START will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.9.7.4 SUBSCRIBE_STOP

Address offset: 0x084

Subscribe configuration for task STOP

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STOP will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.9.7.5 EVENTS_STARTED

Address offset: 0x100

PDM transfer has started

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_STARTED			PDM transfer has started
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.9.7.6 EVENTS_STOPPED

Address offset: 0x104

PDM transfer has finished

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_STOPPED			PDM transfer has finished
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.9.7.7 EVENTS_END

Address offset: 0x108

The PDM has written the last sample specified by SAMPLE.MAXCNT (or the last sample after a STOP task has been received) to Data RAM

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_END			The PDM has written the last sample specified by SAMPLE.MAXCNT (or the last sample after a STOP task has been received) to Data RAM
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.9.7.8 PUBLISH_STARTED

Address offset: 0x180

Publish configuration for event STARTED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event STARTED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.9.7.9 PUBLISH_STOPPED

Address offset: 0x184

Publish configuration for event STOPPED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event STOPPED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.9.7.10 PUBLISH_END

Address offset: 0x188

Publish configuration for event END

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event END will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.9.7.11 INTEN

Address offset: 0x300

Enable or disable interrupt

Bit number	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
ID																														C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	STARTED			Enable or disable interrupt for event STARTED
			Disabled	0	Disable
			Enabled	1	Enable
B	RW	STOPPED			Enable or disable interrupt for event STOPPED
			Disabled	0	Disable
			Enabled	1	Enable
C	RW	END			Enable or disable interrupt for event END
			Disabled	0	Disable
			Enabled	1	Enable

6.9.7.12 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID																														C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	STARTED			Write '1' to enable interrupt for event STARTED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	STOPPED			Write '1' to enable interrupt for event STOPPED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	END			Write '1' to enable interrupt for event END
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.9.7.13 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number				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	
ID				C B A	
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**	
ID	Acce	Field	Value ID	Value	Description
A	RW	STARTED			Write '1' to disable interrupt for event STARTED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	STOPPED			Write '1' to disable interrupt for event STOPPED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	END			Write '1' to disable interrupt for event END
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.9.7.14 ENABLE

Address offset: 0x500

PDM module enable register

Bit number				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	
ID				A	
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**	
ID	Acce	Field	Value ID	Value	Description
A	RW	ENABLE			Enable or disable PDM module
			Disabled	0	Disable
			Enabled	1	Enable

6.9.7.15 PDMCLKCTRL

Address offset: 0x504

PDM clock generator control

Bit number				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	
ID				A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A	
Reset 0x08400000				**0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**	
ID	Acce	Field	Value ID	Value	Description
A	RW	FREQ			PDM_CLK frequency
			1000K	0x08000000	PDM_CLK = 32 MHz / 32 = 1.000 MHz
			Default	0x08400000	PDM_CLK = 32 MHz / 31 = 1.032 MHz. Nominal clock for RATIO=Ratio64.
			1067K	0x08800000	PDM_CLK = 32 MHz / 30 = 1.067 MHz
			1231K	0x09800000	PDM_CLK = 32 MHz / 26 = 1.231 MHz
			1280K	0x0A000000	PDM_CLK = 32 MHz / 25 = 1.280 MHz. Nominal clock for RATIO=Ratio80.
			1333K	0x0A800000	PDM_CLK = 32 MHz / 24 = 1.333 MHz

6.9.7.16 MODE

Address offset: 0x508

Defines the routing of the connected PDM microphones' signals

Bit number	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
ID																															B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	OPERATION			Mono or stereo operation
			Stereo	0	Sample and store one pair (Left + Right) of 16bit samples per RAM word R=[31:16]; L=[15:0]
			Mono	1	Sample and store two successive Left samples (16 bit each) per RAM word L1=[31:16]; L0=[15:0]
B	RW	EDGE			Defines on which PDM_CLK edge Left (or mono) is sampled
			LeftFalling	0	Left (or mono) is sampled on falling edge of PDM_CLK
			LeftRising	1	Left (or mono) is sampled on rising edge of PDM_CLK

6.9.7.17 GAINL

Address offset: 0x518

Left output gain adjustment

Bit number	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
ID																										A	A	A	A	A	A	A
Reset 0x00000028	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	GAINL			Left output gain adjustment, in 0.5 dB steps, around the default module gain (see electrical parameters) 0x00 -20 dB gain adjust 0x01 -19.5 dB gain adjust (...) 0x27 -0.5 dB gain adjust 0x28 0 dB gain adjust 0x29 +0.5 dB gain adjust (...) 0x4F +19.5 dB gain adjust 0x50 +20 dB gain adjust
			MinGain	0x00	-20dB gain adjustment (minimum)
			DefaultGain	0x28	0dB gain adjustment
			MaxGain	0x50	+20dB gain adjustment (maximum)

6.9.7.18 GAINR

Address offset: 0x51C

Right output gain adjustment

Bit number	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
ID																										A	A	A	A	A	A	A
Reset 0x00000028	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	GAINR			Right output gain adjustment, in 0.5 dB steps, around the default module gain (see electrical parameters)
			MinGain	0x00	-20dB gain adjustment (minimum)
			DefaultGain	0x28	0dB gain adjustment
			MaxGain	0x50	+20dB gain adjustment (maximum)

6.9.7.19 RATIO

Address offset: 0x520

Selects the ratio between PDM_CLK and output sample rate. Change PDMCLKCTRL accordingly.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	RATIO			Selects the ratio between PDM_CLK and output sample rate
			Ratio64	0	Ratio of 64
			Ratio80	1	Ratio of 80

6.9.7.20 PSEL.CLK

Address offset: 0x540

Pin number configuration for PDM CLK signal

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.9.7.21 PSEL.DIN

Address offset: 0x544

Pin number configuration for PDM DIN signal

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.9.7.22 SAMPLE.PTR

Address offset: 0x560

RAM address pointer to write samples to with EasyDMA

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	SAMPLEPTR			Address to write PDM samples to over DMA **Note:** See the memory chapter for details about which memories are available for EasyDMA.

6.9.7.23 SAMPLE.MAXCNT

Address offset: 0x564

Number of samples to allocate memory for in EasyDMA mode

Bit number	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
ID																		A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	BUFFSIZE		[0..32767]	Length of DMA RAM allocation in number of samples

6.9.8 Electrical specification

6.9.8.1 PDM Electrical Specification

Symbol	Description	Min.	Typ.	Max.	Units
$f_{PDM,CLK,64}$	PDM clock speed. PDMCLKCTRL = Default (Setting needed for 16MHz sample frequency @ RATIO = Ratio64)		1.032		MHz
$f_{PDM,CLK,80}$	PDM clock speed. PDMCLKCTRL = 1280K (Setting needed for 16MHz sample frequency @ RATIO = Ratio80)	..	..	..	MHz
$t_{PDM,JITTER}$	Jitter in PDM clock output			20	ns
$T_{dPDM,CLK}$	PDM clock duty cycle	40	50	60	%
$t_{PDM,DATA}$	Decimation filter delay			5	ms
$t_{PDM,cv}$	Allowed clock edge to data valid			125	ns
$t_{PDM,ci}$	Allowed (other) clock edge to data invalid	0			ns
$t_{PDM,s}$	Data setup time at $f_{PDM,CLK}$=1.024 MHz or 1.280 MHz	65			ns
$t_{PDM,h}$	Data hold time at $f_{PDM,CLK}$=1.024 MHz or 1.280 MHz	0			ns
$G_{PDM,default}$	Default (reset) absolute gain of the PDM module		3.2		dB

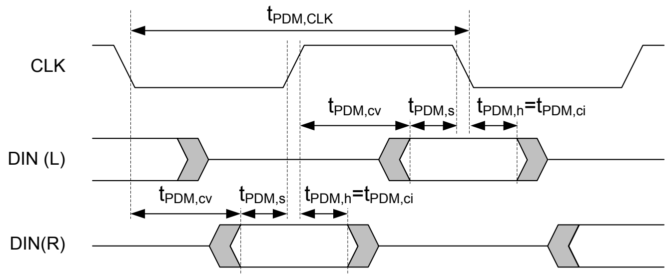

Figure 40: PDM timing diagram

6.10 PWM — Pulse width modulation

The pulse with modulation (PWM) module enables the generation of pulse width modulated signals on GPIO. The module implements an up or up-and-down counter with four PWM channels that drive assigned GPIOs.

The following are the main features of a PWM module:

- Programmable PWM frequency
- Up to four PWM channels with individual polarity and duty cycle values
- Edge or center-aligned pulses across PWM channels
- Multiple duty cycle arrays (sequences) defined in RAM
- Autonomous and glitch-free update of duty cycle values directly from memory through EasyDMA (no CPU involvement)
- Change of polarity, duty cycle, and base frequency possibly on every PWM period
- RAM sequences can be repeated or connected into loops

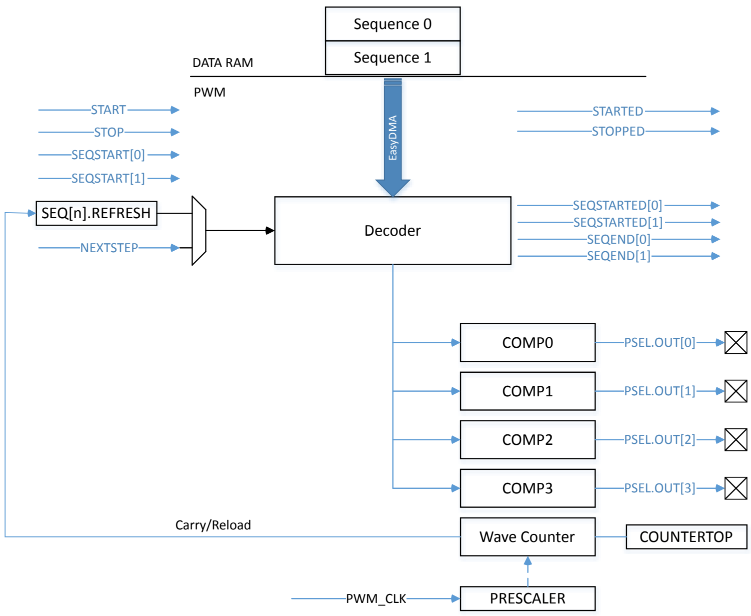

Figure 41: PWM module

6.10.1 Wave counter

The wave counter is responsible for generating the pulses at a duty cycle that depends on the compare values, and at a frequency that depends on COUNTERTOP.

There is one common 15-bit counter with four compare channels. Thus, all four channels will share the same period (PWM frequency), but can have individual duty cycle and polarity. The polarity is set by a value read from RAM (see figure Decoder memory access modes on page 165). Whether the counter counts up, or up and down, is controlled by the MODE register.

The timer top value is controlled by the COUNTERTOP register. This register value, in conjunction with the selected PRESCALER of the PWM_CLK, will result in a given PWM period. A COUNTERTOP value smaller than the compare setting will result in a state where no PWM edges are generated. OUT[n] is held high, given that the polarity is set to FallingEdge. All compare registers are internal and can only be configured through decoder presented later. COUNTERTOP can be safely written at any time.

Sampling follows the START task. If DECODER.LOAD=WaveForm, the register value is ignored and taken from RAM instead (see section Decoder with EasyDMA on page 165 for more details). If DECODER.LOAD is anything else than the WaveForm, it is sampled following a STARTSEQ[n] task and when loading a new value from RAM during a sequence playback.

The following figure shows the counter operating in up mode (MODE=PWM_MODE_Up), with three PWM channels with the same frequency but different duty cycle:

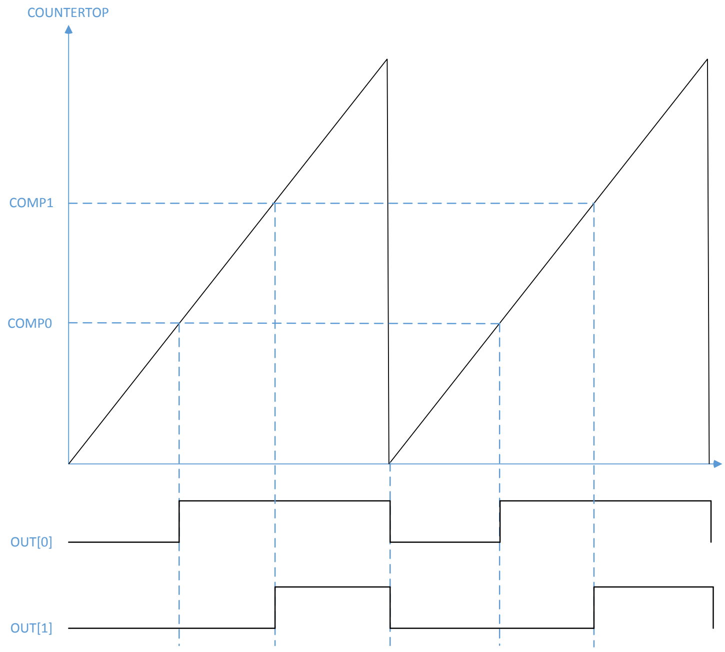

Figure 42: PWM counter in up mode example - FallingEdge polarity

The counter is automatically reset to zero when COUNTERTOP is reached and OUT[n] will invert. OUT[n] is held low if the compare value is 0 and held high if set to COUNTERTOP, given that the polarity is set to

FallingEdge. Counter running in up mode results in pulse widths that are edge-aligned. The following is the code for the counter in up mode example:

```
uint16_t pwm_seq[4] = {PWM_CH0_DUTY, PWM_CH1_DUTY, PWM_CH2_DUTY, PWM_CH3_DUTY};
NRF_PWM0->PSEL.OUT[0] = (first_pin << PWM_PSEL_OUT_PIN_Pos) |
                        (PWM_PSEL_OUT_CONNECT_Connected <<
                                                 PWM_PSEL_OUT_CONNECT_Pos);
NRF_PWM0->PSEL.OUT[1] = (second_pin << PWM_PSEL_OUT_PIN_Pos) |
                        (PWM_PSEL_OUT_CONNECT_Connected <<
                                                 PWM_PSEL_OUT_CONNECT_Pos);
NRF_PWM0->ENABLE      = (PWM_ENABLE_ENABLE_Enabled << PWM_ENABLE_ENABLE_Pos);
NRF_PWM0->MODE        = (PWM_MODE_UPDOWN_Up << PWM_MODE_UPDOWN_Pos);
NRF_PWM0->PRESCALER   = (PWM_PRESCALER_PRESCALER_DIV_1 <<
                                                 PWM_PRESCALER_PRESCALER_Pos);
NRF_PWM0->COUNTERTOP  = (16000 << PWM_COUNTERTOP_COUNTERTOP_Pos); //1 msec
NRF_PWM0->LOOP        = (PWM_LOOP_CNT_Disabled << PWM_LOOP_CNT_Pos);
NRF_PWM0->DECODER   = (PWM_DECODER_LOAD_Individual << PWM_DECODER_LOAD_Pos) |
                      (PWM_DECODER_MODE_RefreshCount << PWM_DECODER_MODE_Pos);
NRF_PWM0->SEQ[0].PTR  = ((uint32_t)(pwm_seq) << PWM_SEQ_PTR_PTR_Pos);
NRF_PWM0->SEQ[0].CNT  = ((sizeof(pwm_seq) / sizeof(uint16_t)) <<
                                                 PWM_SEQ_CNT_CNT_Pos);
NRF_PWM0->SEQ[0].REFRESH  = 0;
NRF_PWM0->SEQ[0].ENDDELAY = 0;
NRF_PWM0->TASKS_SEQSTART[0] = 1;
```

When the counter is running in up mode, the following formula can be used to compute the PWM period and the step size:

PWM period: $T_{PWM(Up)} = T_{PWM\_CLK} * COUNTERTOP$

Step width/Resolution: $T_{steps} = T_{PWM\_CLK}$

The following figure shows the counter operating in up-and-down mode (MODE=PWM_MODE_UpAndDown), with two PWM channels with the same frequency but different duty cycle and output polarity:

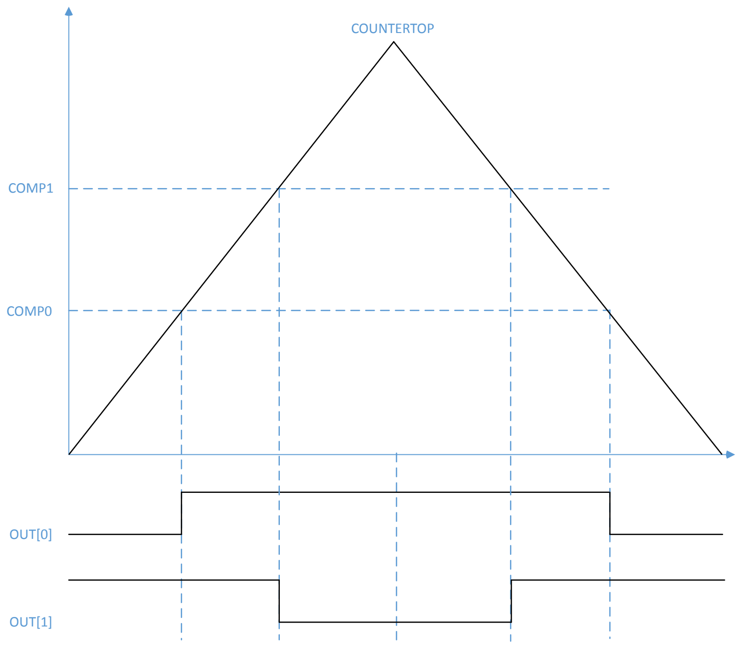

Figure 43: PWM counter in up-and-down mode example

The counter starts decrementing to zero when COUNTERTOP is reached and will invert the OUT[n] when compare value is hit for the second time. This results in a set of pulses that are center-aligned. The following is the code for the counter in up-and-down mode example:

```
uint16_t pwm_seq[4] = {PWM_CH0_DUTY, PWM_CH1_DUTY, PWM_CH2_DUTY, PWM_CH3_DUTY};
NRF_PWM0->PSEL.OUT[0] = (first_pin << PWM_PSEL_OUT_PIN_Pos) |
                        (PWM_PSEL_OUT_CONNECT_Connected <<
                                                 PWM_PSEL_OUT_CONNECT_Pos);
NRF_PWM0->PSEL.OUT[1] = (second_pin << PWM_PSEL_OUT_PIN_Pos) |
                        (PWM_PSEL_OUT_CONNECT_Connected <<
                                                 PWM_PSEL_OUT_CONNECT_Pos);
NRF_PWM0->ENABLE      = (PWM_ENABLE_ENABLE_Enabled << PWM_ENABLE_ENABLE_Pos);
NRF_PWM0->MODE        = (PWM_MODE_UPDOWN_UpAndDown << PWM_MODE_UPDOWN_Pos);
NRF_PWM0->PRESCALER   = (PWM_PRESCALER_PRESCALER_DIV_1 <<
                                                 PWM_PRESCALER_PRESCALER_Pos);
NRF_PWM0->COUNTERTOP  = (16000 << PWM_COUNTERTOP_COUNTERTOP_Pos); //1 msec
NRF_PWM0->LOOP        = (PWM_LOOP_CNT_Disabled << PWM_LOOP_CNT_Pos);
NRF_PWM0->DECODER    = (PWM_DECODER_LOAD_Individual << PWM_DECODER_LOAD_Pos) |
                      (PWM_DECODER_MODE_RefreshCount << PWM_DECODER_MODE_Pos);
NRF_PWM0->SEQ[0].PTR  = ((uint32_t)(pwm_seq) << PWM_SEQ_PTR_PTR_Pos);
NRF_PWM0->SEQ[0].CNT  = ((sizeof(pwm_seq) / sizeof(uint16_t)) <<
                                                 PWM_SEQ_CNT_CNT_Pos);
NRF_PWM0->SEQ[0].REFRESH  = 0;
NRF_PWM0->SEQ[0].ENDDELAY = 0;
NRF_PWM0->TASKS_SEQSTART[0] = 1;
```

When the counter is running in up-and-down mode, the following formula can be used to compute the PWM period and the step size:

$T_{PWM(Up\ And\ Down)} = T_{PWM\_CLK} * 2 * COUNTERTOP$

Step width/Resolution: $T_{steps} = T_{PWM\_CLK} * 2$

6.10.2 Decoder with EasyDMA

The decoder uses EasyDMA to take PWM parameters stored in RAM and update the internal compare registers of the wave counter, based on the mode of operation.

PWM parameters are organized into a sequence containing at least one half word (16 bit). Its most significant bit[15] denotes the polarity of the OUT[n] while bit[14:0] is the 15-bit compare value.

Bit number	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
Id																	B	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Id	RW	Field	Value Id	Value	Description
A	RW	COMPARE			Duty cycle setting - value loaded to internal compare register
B	RW	POLARITY			Edge polarity of GPIO.
			RisingEdge	0	First edge within the PWM period is rising
			FallingEdge	1	First edge within the PWM period is falling

The DECODER register controls how the RAM content is interpreted and loaded into the internal compare registers. The LOAD field controls if the RAM values are loaded to all compare channels, or to update a group or all channels with individual values. The following figure illustrates how parameters stored in RAM are organized and routed to various compare channels in different modes:

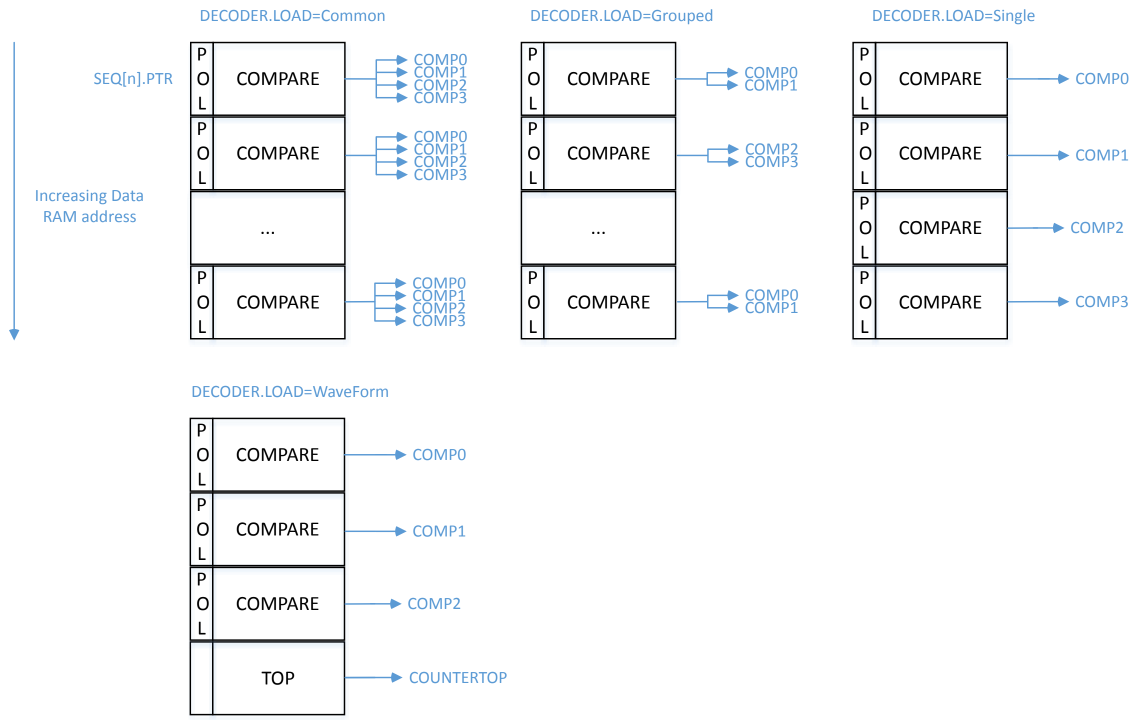

Figure 44: Decoder memory access modes

A special mode of operation is available when DECODER.LOAD is set to WaveForm. In this mode, up to three PWM channels can be enabled - OUT[0] to OUT[2]. In RAM, four values are loaded at a time: the first, second and third location are used to load the values, and the fourth RAM location is used to load the COUNTERTOP register. This way one can have up to three PWM channels with a frequency base that changes on a per PWM period basis. This mode of operation is useful for arbitrary wave form generation in applications, such as LED lighting.

The register SEQ[n].REFRESH=N (one per sequence n=0 or 1) will instruct a new RAM stored pulse width value on every (N+1)th PWM period. Setting the register to zero will result in a new duty cycle update every PWM period, as long as the minimum PWM period is observed.

Note that registers SEQ[n].REFRESH and SEQ[n].ENDDELAY are ignored when DECODER.MODE=NextStep. The next value is loaded upon every received NEXTSTEP task.

SEQ[n].PTR is the pointer used to fetch COMPARE values from RAM. If the SEQ[n].PTR is not pointing to a RAM region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 20 for more information about the different memory regions. After the SEQ[n].PTR is set to the desired RAM location, the SEQ[n].CNT register must be set to number of 16-bit half words in the sequence. It is important to observe that the Grouped mode requires one half word per group, while the Single mode requires one half word per channel, thus increasing the RAM size occupation. If PWM generation is not running when the SEQSTART[n] task is triggered, the task will load the first value from RAM and then start the PWM generation. A SEQSTARTED[n] event is generated as soon as the EasyDMA has read the first PWM parameter from RAM and the wave counter has started executing it. When LOOP.CNT=0, sequence n=0 or 1 is played back once. After the last value in the sequence has been loaded and started executing, a SEQEND[n] event is generated. The PWM generation will then continue with the last loaded value. The following figure illustrates an example of such simple playback:

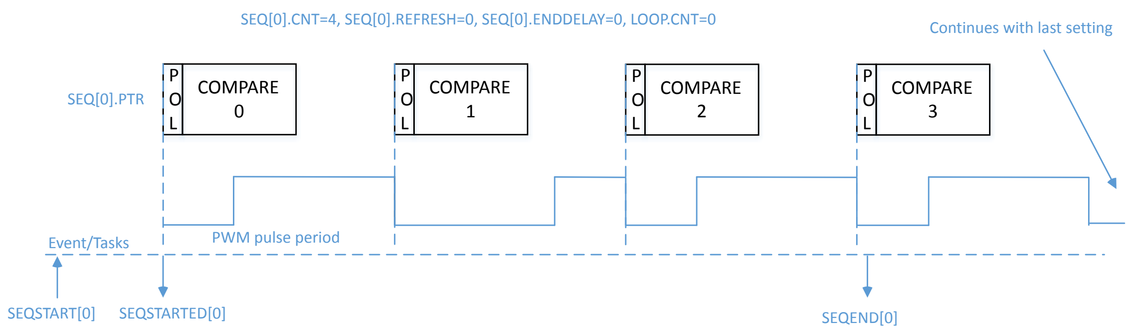

Figure 45: Simple sequence example

Figure depicts the source code used for configuration and timing details in a sequence where only sequence 0 is used and only run once with a new PWM duty cycle for each period.

```
NRF_PWM0->PSEL.OUT[0] = (first_pin << PWM_PSEL_OUT_PIN_Pos) |
                        (PWM_PSEL_OUT_CONNECT_Connected <<
                                                 PWM_PSEL_OUT_CONNECT_Pos);
NRF_PWM0->ENABLE      = (PWM_ENABLE_ENABLE_Enabled << PWM_ENABLE_ENABLE_Pos);
NRF_PWM0->MODE        = (PWM_MODE_UPDOWN_Up << PWM_MODE_UPDOWN_Pos);
NRF_PWM0->PRESCALER   = (PWM_PRESCALER_PRESCALER_DIV_1 <<
                                                 PWM_PRESCALER_PRESCALER_Pos);
NRF_PWM0->COUNTERTOP  = (16000 << PWM_COUNTERTOP_COUNTERTOP_Pos); //1 msec
NRF_PWM0->LOOP        = (PWM_LOOP_CNT_Disabled << PWM_LOOP_CNT_Pos);
NRF_PWM0->DECODER   = (PWM_DECODER_LOAD_Common << PWM_DECODER_LOAD_Pos) |
                      (PWM_DECODER_MODE_RefreshCount << PWM_DECODER_MODE_Pos);
NRF_PWM0->SEQ[0].PTR  = ((uint32_t)(seq0_ram) << PWM_SEQ_PTR_PTR_Pos);
NRF_PWM0->SEQ[0].CNT  = ((sizeof(seq0_ram) / sizeof(uint16_t)) <<
                                                 PWM_SEQ_CNT_CNT_Pos);
NRF_PWM0->SEQ[0].REFRESH  = 0;
NRF_PWM0->SEQ[0].ENDDELAY = 0;
NRF_PWM0->TASKS_SEQSTART[0] = 1;
```

To completely stop the PWM generation and force the associated pins to a defined state, a STOP task can be triggered at any time. A STOPPED event is generated when the PWM generation has stopped at the end of currently running PWM period, and the pins go into their idle state as defined in GPIO OUT register. PWM generation can then only be restarted through a SEQSTART[n] task. SEQSTART[n] will resume PWM generation after having loaded the first value from the RAM buffer defined in the SEQ[n].PTR register.

The table below indicates when specific registers get sampled by the hardware. Care should be taken when updating these registers to avoid that values are applied earlier than expected.

Register	Taken into account by hardware	Recommended (safe) update
SEQ[n].PTR	When sending the SEQSTART[n] task	After having received the SEQSTARTED[n] event
SEQ[n].CNT	When sending the SEQSTART[n] task	After having received the SEQSTARTED[n] event
SEQ[0].ENDDELAY	When sending the SEQSTART[0] task Every time a new value from sequence [0] has been loaded from RAM and gets applied to the Wave Counter (indicated by the PWMPERIODEND event)	Before starting sequence [0] through a SEQSTART[0] task When no more value from sequence [0] gets loaded from RAM (indicated by the SEQEND[0] event) At any time during sequence [1] (which starts when the SEQSTARTED[1] event is generated)
SEQ[1].ENDDELAY	When sending the SEQSTART[1] task Every time a new value from sequence [1] has been loaded from RAM and gets applied to the Wave Counter (indicated by the PWMPERIODEND event)	Before starting sequence [1] through a SEQSTART[1] task When no more value from sequence [1] gets loaded from RAM (indicated by the SEQEND[1] event) At any time during sequence [0] (which starts when the SEQSTARTED[0] event is generated)
SEQ[0].REFRESH	When sending the SEQSTART[0] task Every time a new value from sequence [0] has been loaded from RAM and gets applied to the Wave Counter (indicated by the PWMPERIODEND event)	Before starting sequence [0] through a SEQSTART[0] task At any time during sequence [1] (which starts when the SEQSTARTED[1] event is generated)
SEQ[1].REFRESH	When sending the SEQSTART[1] task Every time a new value from sequence [1] has been loaded from RAM and gets applied to the Wave Counter (indicated by the PWMPERIODEND event)	Before starting sequence [1] through a SEQSTART[1] task At any time during sequence [0] (which starts when the SEQSTARTED[0] event is generated)
COUNTERTOP	In DECODER.LOAD=WaveForm: this register is ignored. In all other LOAD modes: at the end of current PWM period (indicated by the PWMPERIODEND event)	Before starting PWM generation through a SEQSTART[n] task After a STOP task has been triggered, and the STOPPED event has been received.
MODE	Immediately	Before starting PWM generation through a SEQSTART[n] task After a STOP task has been triggered, and the STOPPED event has been received.
DECODER	Immediately	Before starting PWM generation through a SEQSTART[n] task After a STOP task has been triggered, and the STOPPED event has been received.
PRESCALER	Immediately	Before starting PWM generation through a SEQSTART[n] task After a STOP task has been triggered, and the STOPPED event has been received.
LOOP	Immediately	Before starting PWM generation through a SEQSTART[n] task After a STOP task has been triggered, and the STOPPED event has been received.
PSEL.OUT[n]	Immediately	Before enabling the PWM instance through the ENABLE register

Table 57: When to safely update PWM registers

Note: SEQ[n].REFRESH and SEQ[n].ENDDELAY are ignored at the end of a complex sequence, indicated by a LOOPSDONE event. The reason for this is that the last value loaded from RAM is maintained until further action from software (restarting a new sequence, or stopping PWM generation).

A more complex example, where LOOP.CNT>0, is shown in the following figure:

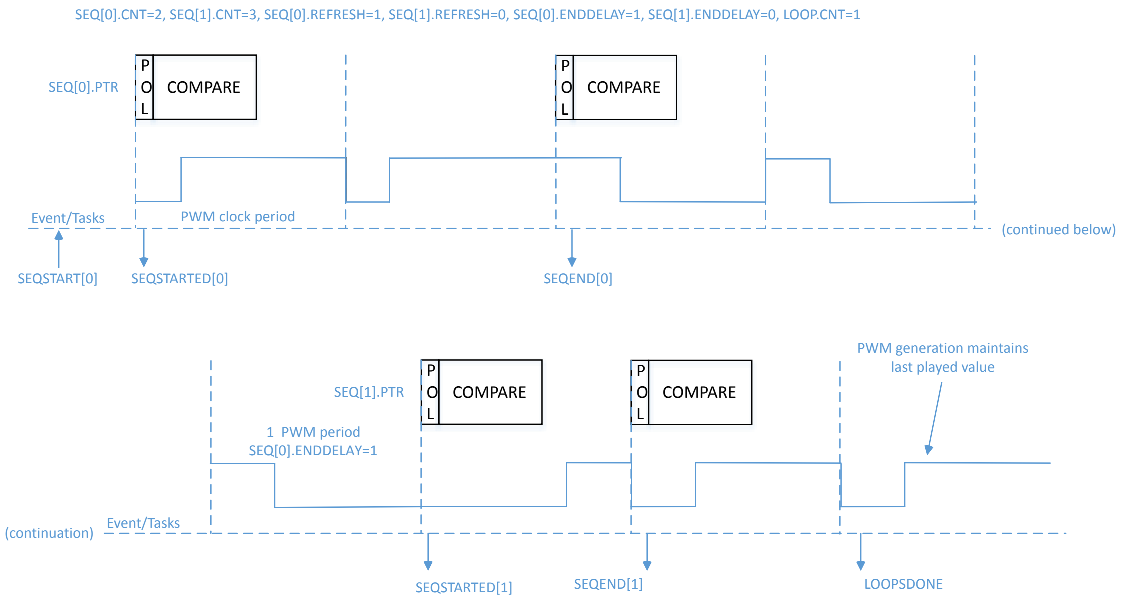

Figure 46: Example using two sequences

In this case, an automated playback takes place, consisting of SEQ[0], delay 0, SEQ[1], delay 1, then again SEQ[0], etc. The user can choose to start a complex playback with SEQ[0] or SEQ[1] through sending the SEQSTART[0] or SEQSTART[1] task. The complex playback always ends with delay 1.

The two sequences 0 and 1 are defined by the addresses of value tables in RAM (pointed to by SEQ[n].PTR) and the buffer size (SEQ[n].CNT). The rate at which a new value is loaded is defined individually for each sequence by SEQ[n].REFRESH. The chaining of sequence 1 following the sequence 0 is implicit, the LOOP.CNT register allows the chaining of sequence 1 to sequence 0 for a determined number of times. In other words, it allows to repeat a complex sequence a number of times in a fully automated way.

In the following code example, sequence 0 is defined with SEQ[0].REFRESH set to 1, meaning that a new PWM duty cycle is pushed every second PWM period. This complex sequence is started with the SEQSTART[0] task, so SEQ[0] is played first. Since SEQ[0].ENDDELAY=1 there will be one PWM period delay between last period on sequence 0 and the first period on sequence 1. Since SEQ[1].ENDDELAY=0 there is no delay 1, so SEQ[0] would be started immediately after the end of SEQ[1]. However, as LOOP.CNT is

1, the playback stops after having played SEQ[1] only once, and both SEQEND[1] and LOOPSDONE are generated (their order is not guaranteed in this case).

```
NRF_PWM0->PSEL.OUT[0] = (first_pin << PWM_PSEL_OUT_PIN_Pos) |
                        (PWM_PSEL_OUT_CONNECT_Connected <<
                                                 PWM_PSEL_OUT_CONNECT_Pos);
NRF_PWM0->ENABLE      = (PWM_ENABLE_ENABLE_Enabled << PWM_ENABLE_ENABLE_Pos);
NRF_PWM0->MODE        = (PWM_MODE_UPDOWN_Up << PWM_MODE_UPDOWN_Pos);
NRF_PWM0->PRESCALER   = (PWM_PRESCALER_PRESCALER_DIV_1 <<
                                                 PWM_PRESCALER_PRESCALER_Pos);
NRF_PWM0->COUNTERTOP  = (16000 << PWM_COUNTERTOP_COUNTERTOP_Pos); //1 msec
NRF_PWM0->LOOP        = (1 << PWM_LOOP_CNT_Pos);
NRF_PWM0->DECODER   = (PWM_DECODER_LOAD_Common << PWM_DECODER_LOAD_Pos) |
                      (PWM_DECODER_MODE_RefreshCount << PWM_DECODER_MODE_Pos);
NRF_PWM0->SEQ[0].PTR  = ((uint32_t)(seq0_ram) << PWM_SEQ_PTR_PTR_Pos);
NRF_PWM0->SEQ[0].CNT  = ((sizeof(seq0_ram) / sizeof(uint16_t)) <<
                                                 PWM_SEQ_CNT_CNT_Pos);
NRF_PWM0->SEQ[0].REFRESH  = 1;
NRF_PWM0->SEQ[0].ENDDELAY = 1;
NRF_PWM0->SEQ[1].PTR  = ((uint32_t)(seq1_ram) << PWM_SEQ_PTR_PTR_Pos);
NRF_PWM0->SEQ[1].CNT  = ((sizeof(seq1_ram) / sizeof(uint16_t)) <<
                                                 PWM_SEQ_CNT_CNT_Pos);
NRF_PWM0->SEQ[1].REFRESH  = 0;
NRF_PWM0->SEQ[1].ENDDELAY = 0;
NRF_PWM0->TASKS_SEQSTART[0] = 1;
```

The decoder can also be configured to asynchronously load new PWM duty cycle. If the DECODER.MODE register is set to NextStep, then the NEXTSTEP task will cause an update of internal compare registers on the next PWM period.

The following figures provide an overview of each part of an arbitrary sequence, in various modes (LOOP.CNT=0 and LOOP.CNT>0). In particular, the following are represented:

- Initial and final duty cycle on the PWM output(s)
- Chaining of SEQ[0] and SEQ[1] if LOOP.CNT>0
- Influence of registers on the sequence
- Events generated during a sequence
- DMA activity (loading of next value and applying it to the output(s))

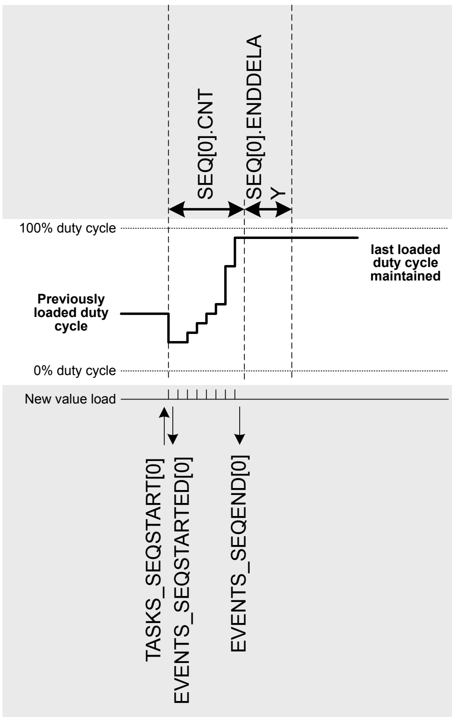

Figure 47: Single shot (LOOP.CNT=0)

Note: The single-shot example also applies to SEQ[1]. Only SEQ[0] is represented for simplicity.

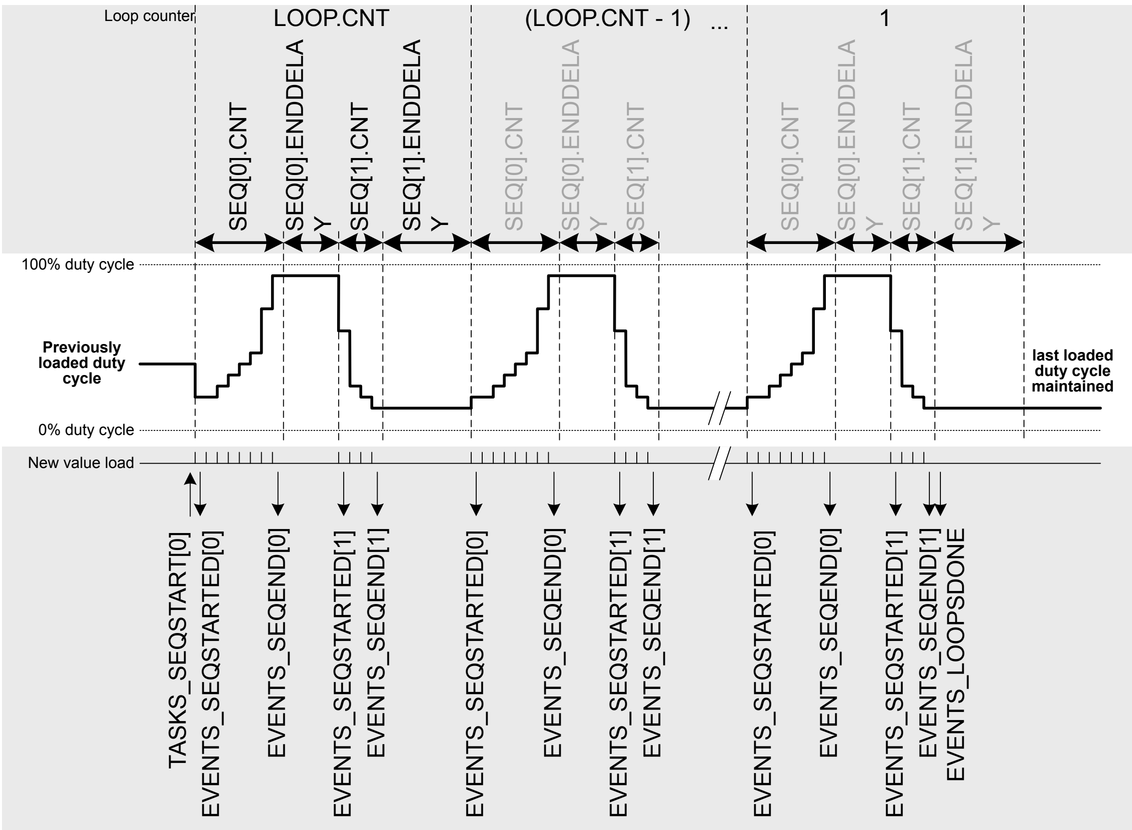

Figure 48: Complex sequence (LOOP.CNT>0) starting with SEQ[0]

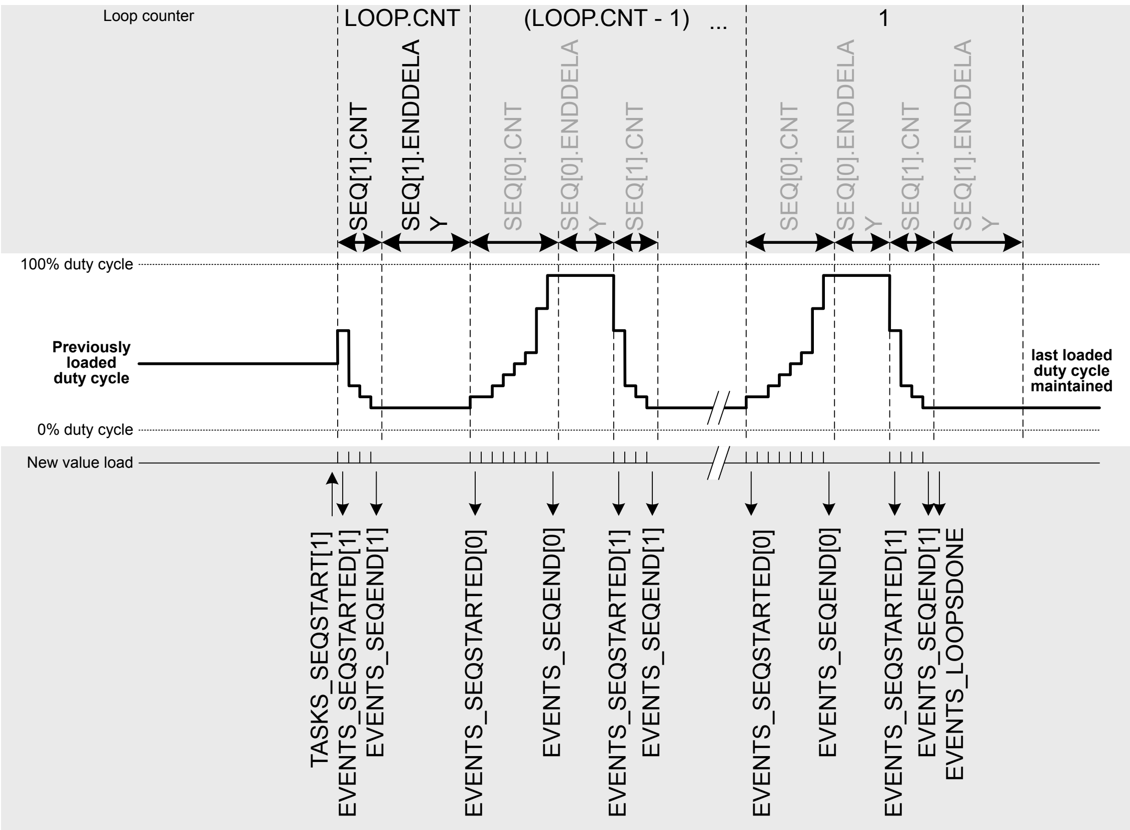

Figure 49: Complex sequence (LOOP.CNT>0) starting with SEQ[1]

Note: If a sequence is in use in a simple or complex sequence, it must have a length of SEQ[n].CNT > 0.

6.10.3 Limitations

Previous compare value is repeated if the PWM period is shorter than the time it takes for the EasyDMA to retrieve from RAM and update the internal compare registers. This is to ensure a glitch-free operation even for very short PWM periods.

6.10.4 Pin configuration

The OUT[n] (n=0..3) signals associated with each PWM channel are mapped to physical pins according to the configuration of PSEL.OUT[n] registers. If PSEL.OUT[n].CONNECT is set to Disconnected, the associated PWM module signal will not be connected to any physical pins.

The PSEL.OUT[n] registers and their configurations are used as long as the PWM module is enabled and the PWM generation active (wave counter started). They are retained only as long as the device is in System ON mode (see section POWER for more information about power modes).

To ensure correct behavior in the PWM module, the pins that are used must be configured in the GPIO peripheral in the following way before the PWM module is enabled:

PWM signal	PWM pin	Direction	Output value	Comment
OUT[n]	As specified in PSEL.OUT[n] (n=0..3)	Output	0	Idle state defined in GPIO OUT register

Table 58: Recommended GPIO configuration before starting PWM generation

The idle state of a pin is defined by the OUT register in the GPIO module, to ensure that the pins used by the PWM module are driven correctly. If PWM generation is stopped by triggering a STOP task, the PWM module itself is temporarily disabled or the device temporarily enters System OFF. This configuration must be retained in the GPIO for the selected pins (I/Os) for as long as the PWM module is supposed to be connected to an external PWM circuit.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

6.10.5 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50021000 0x40021000	PWM	PWM0 : S PWM0 : NS	US	SA	Pulse width modulation unit 0	
0x50022000 0x40022000	PWM	PWM1 : S PWM1 : NS	US	SA	Pulse width modulation unit 1	
0x50023000 0x40023000	PWM	PWM2 : S PWM2 : NS	US	SA	Pulse width modulation unit 2	
0x50024000 0x40024000	PWM	PWM3 : S PWM3 : NS	US	SA	Pulse width modulation unit 3	

Table 59: Instances

Register	Offset	Security	Description
TASKS_STOP	0x004		Stops PWM pulse generation on all channels at the end of current PWM period, and stops sequence playback
TASKS_SEQSTART[0]	0x008		Loads the first PWM value on all enabled channels from sequence 0, and starts playing that sequence at the rate defined in SEQ[0]REFRESH and/or DECODER.MODE. Causes PWM generation to start if not running.
TASKS_SEQSTART[1]	0x00C		Loads the first PWM value on all enabled channels from sequence 1, and starts playing that sequence at the rate defined in SEQ[1]REFRESH and/or DECODER.MODE. Causes PWM generation to start if not running.
TASKS_NEXTSTEP	0x010		Steps by one value in the current sequence on all enabled channels if DECODER.MODE=NextStep. Does not cause PWM generation to start if not running.
SUBSCRIBE_STOP	0x084		Subscribe configuration for task STOP
SUBSCRIBE_SEQSTART[0]	0x088		Subscribe configuration for task SEQSTART[0]
SUBSCRIBE_SEQSTART[1]	0x08C		Subscribe configuration for task SEQSTART[1]
SUBSCRIBE_NEXTSTEP	0x090		Subscribe configuration for task NEXTSTEP
EVENTS_STOPPED	0x104		Response to STOP task, emitted when PWM pulses are no longer generated
EVENTS_SEQSTARTED[0]	0x108		First PWM period started on sequence 0
EVENTS_SEQSTARTED[1]	0x10C		First PWM period started on sequence 1
EVENTS_SEQEND[0]	0x110		Emitted at end of every sequence 0, when last value from RAM has been applied to wave counter
EVENTS_SEQEND[1]	0x114		Emitted at end of every sequence 1, when last value from RAM has been applied to wave counter
EVENTS_PWMPERIODEND	0x118		Emitted at the end of each PWM period
EVENTS_LOOPSDONE	0x11C		Concatenated sequences have been played the amount of times defined in LOOP.CNT
PUBLISH_STOPPED	0x184		Publish configuration for event STOPPED
PUBLISH_SEQSTARTED[0]	0x188		Publish configuration for event SEQSTARTED[0]
PUBLISH_SEQSTARTED[1]	0x18C		Publish configuration for event SEQSTARTED[1]
PUBLISH_SEQEND[0]	0x190		Publish configuration for event SEQEND[0]
PUBLISH_SEQEND[1]	0x194		Publish configuration for event SEQEND[1]

Register	Offset	Security	Description
PUBLISH_PWMPERIODEND	0x198		Publish configuration for event PWMPERIODEND
PUBLISH_LOOPSDONE	0x19C		Publish configuration for event LOOPSDONE
SHORTS	0x200		Shortcuts between local events and tasks
INTEN	0x300		Enable or disable interrupt
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
ENABLE	0x500		PWM module enable register
MODE	0x504		Selects operating mode of the wave counter
COUNTERTOP	0x508		Value up to which the pulse generator counter counts
PRESCALER	0x50C		Configuration for PWM_CLK
DECODER	0x510		Configuration of the decoder
LOOP	0x514		Number of playbacks of a loop
SEQ[0].PTR	0x520		Beginning address in RAM of this sequence
SEQ[0].CNT	0x524		Number of values (duty cycles) in this sequence
SEQ[0].REFRESH	0x528		Number of additional PWM periods between samples loaded into compare register
SEQ[0].ENDDELAY	0x52C		Time added after the sequence
SEQ[1].PTR	0x540		Beginning address in RAM of this sequence
SEQ[1].CNT	0x544		Number of values (duty cycles) in this sequence
SEQ[1].REFRESH	0x548		Number of additional PWM periods between samples loaded into compare register
SEQ[1].ENDDELAY	0x54C		Time added after the sequence
PSEL.OUT[0]	0x560		Output pin select for PWM channel 0
PSEL.OUT[1]	0x564		Output pin select for PWM channel 1
PSEL.OUT[2]	0x568		Output pin select for PWM channel 2
PSEL.OUT[3]	0x56C		Output pin select for PWM channel 3

Table 60: Register overview

6.10.5.1 TASKS_STOP

Address offset: 0x004

Stops PWM pulse generation on all channels at the end of current PWM period, and stops sequence playback

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_STOP			Stops PWM pulse generation on all channels at the end of current PWM period, and stops sequence playback
			Trigger	1	Trigger task

6.10.5.2 TASKS_SEQSTART[n] (n=0..1)

Address offset: 0x008 + (n × 0x4)

Loads the first PWM value on all enabled channels from sequence n, and starts playing that sequence at the rate defined in SEQ[n]REFRESH and/or DECODER.MODE. Causes PWM generation to start if not running.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_SEQSTART			Loads the first PWM value on all enabled channels from sequence n, and starts playing that sequence at the rate defined in SEQ[n]REFRESH and/or DECODER.MODE. Causes PWM generation to start if not running.
			Trigger	1	Trigger task

6.10.5.3 TASKS_NEXTSTEP

Address offset: 0x010

Steps by one value in the current sequence on all enabled channels if DECODER.MODE=NextStep. Does not cause PWM generation to start if not running.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_NEXTSTEP			Steps by one value in the current sequence on all enabled channels if DECODER.MODE=NextStep. Does not cause PWM generation to start if not running.
			Trigger	1	Trigger task

6.10.5.4 SUBSCRIBE_STOP

Address offset: 0x084

Subscribe configuration for task STOP

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STOP will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.10.5.5 SUBSCRIBE_SEQSTART[n] (n=0..1)

Address offset: 0x088 + (n × 0x4)

Subscribe configuration for task SEQSTART[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task SEQSTART[n] will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.10.5.6 SUBSCRIBE_NEXTSTEP

Address offset: 0x090

Subscribe configuration for task NEXTSTEP

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task NEXTSTEP will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.10.5.7 EVENTS_STOPPED

Address offset: 0x104

Response to STOP task, emitted when PWM pulses are no longer generated

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_STOPPED			Response to STOP task, emitted when PWM pulses are no longer generated
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.10.5.8 EVENTS_SEQSTARTED[n] (n=0..1)

Address offset: 0x108 + (n × 0x4)

First PWM period started on sequence n

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_SEQSTARTED			First PWM period started on sequence n
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.10.5.9 EVENTS_SEQEND[n] (n=0..1)

Address offset: 0x110 + (n × 0x4)

Emitted at end of every sequence n, when last value from RAM has been applied to wave counter

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_SEQEND			Emitted at end of every sequence n, when last value from RAM has been applied to wave counter
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.10.5.10 EVENTS_PWMPERIODEND

Address offset: 0x118

Emitted at the end of each PWM period

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_PWMPERIODEND			Emitted at the end of each PWM period
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.10.5.11 EVENTS_LOOPSDONE

Address offset: 0x11C

Concatenated sequences have been played the amount of times defined in LOOP.CNT

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_LOOPSDONE			Concatenated sequences have been played the amount of times defined in LOOP.CNT
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.10.5.12 PUBLISH_STOPPED

Address offset: 0x184

Publish configuration for event STOPPED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event STOPPED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.10.5.13 PUBLISH_SEQSTARTED[n] (n=0..1)

Address offset: 0x188 + (n × 0x4)

Publish configuration for event SEQSTARTED[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event SEQSTARTED[n] will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.10.5.14 PUBLISH_SEQEND[n] (n=0..1)

Address offset: 0x190 + (n × 0x4)

Publish configuration for event SEQEND[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event SEQEND[n] will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.10.5.15 PUBLISH_PWMPERIODEND

Address offset: 0x198

Publish configuration for event PWMPERIODEND

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event PWMPERIODEND will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.10.5.16 PUBLISH_LOOPSDONE

Address offset: 0x19C

Publish configuration for event LOOPSDONE

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event LOOPSDONE will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.10.5.17 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

Bit number	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
ID																												E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	SEQEND0_STOP			Shortcut between event SEQEND[0] and task STOP
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut
B	RW	SEQEND1_STOP			Shortcut between event SEQEND[1] and task STOP
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut
C	RW	LOOPSDONE_SEQSTART0			Shortcut between event LOOPSDONE and task SEQSTART[0]
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut
D	RW	LOOPSDONE_SEQSTART1			Shortcut between event LOOPSDONE and task SEQSTART[1]
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut
E	RW	LOOPSDONE_STOP			Shortcut between event LOOPSDONE and task STOP
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut

6.10.5.18 INTEN

Address offset: 0x300

Enable or disable interrupt

Bit number	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
ID																									H	G	F	E	D	C	B	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
B	RW	STOPPED			Enable or disable interrupt for event STOPPED
			Disabled	0	Disable
			Enabled	1	Enable
C-D	RW	SEQSTARTED[i] (i=0..1)			Enable or disable interrupt for event SEQSTARTED[i]
			Disabled	0	Disable
			Enabled	1	Enable
E-F	RW	SEQEND[i] (i=0..1)			Enable or disable interrupt for event SEQEND[i]
			Disabled	0	Disable
			Enabled	1	Enable
G	RW	PWMPERIODEND			Enable or disable interrupt for event PWMPERIODEND

Bit number	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
ID																									H	G	F	E	D	C	B	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			Disabled	0	Disable
			Enabled	1	Enable
H	RW	LOOPSDONE			Enable or disable interrupt for event LOOPSDONE
			Disabled	0	Disable
			Enabled	1	Enable

6.10.5.19 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID																									H	G	F	E	D	C	B	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
B	RW	STOPPED			Write '1' to enable interrupt for event STOPPED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C-D	RW	SEQSTARTED[i] (i=0..1)			Write '1' to enable interrupt for event SEQSTARTED[i]
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
E-F	RW	SEQEND[i] (i=0..1)			Write '1' to enable interrupt for event SEQEND[i]
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
G	RW	PWMPERIODEND			Write '1' to enable interrupt for event PWMPERIODEND
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
H	RW	LOOPSDONE			Write '1' to enable interrupt for event LOOPSDONE
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.10.5.20 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID																									H	G	F	E	D	C	B	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
B	RW	STOPPED			Write '1' to disable interrupt for event STOPPED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

Bit number	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
ID																									H	G	F	E	D	C	B	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
C-D	RW	SEQSTARTED[i] (i=0..1)			Write '1' to disable interrupt for event SEQSTARTED[i]
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
E-F	RW	SEQEND[i] (i=0..1)			Write '1' to disable interrupt for event SEQEND[i]
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
G	RW	PWMPERIODEND			Write '1' to disable interrupt for event PWMPERIODEND
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
H	RW	LOOPSDONE			Write '1' to disable interrupt for event LOOPSDONE
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.10.5.21 ENABLE

Address offset: 0x500

PWM module enable register

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ENABLE			Enable or disable PWM module
			Disabled	0	Disabled
			Enabled	1	Enable

6.10.5.22 MODE

Address offset: 0x504

Selects operating mode of the wave counter

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	UPDOWN			Selects up mode or up-and-down mode for the counter
			Up	0	Up counter, edge-aligned PWM duty cycle
			UpAndDown	1	Up and down counter, center-aligned PWM duty cycle

6.10.5.23 COUNTERTOP

Address offset: 0x508

Value up to which the pulse generator counter counts

Bit number	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
ID																		A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x000003FF	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	COUNTERTOP		[3..32767]	Value up to which the pulse generator counter counts. This register is ignored when DECODER.MODE=WaveForm and only values from RAM are used.

6.10.5.24 PRESCALER

Address offset: 0x50C

Configuration for PWM_CLK

Bit number	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
ID																														A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PRESCALER			Prescaler of PWM_CLK
			DIV_1	0	Divide by 1 (16 MHz)
			DIV_2	1	Divide by 2 (8 MHz)
			DIV_4	2	Divide by 4 (4 MHz)
			DIV_8	3	Divide by 8 (2 MHz)
			DIV_16	4	Divide by 16 (1 MHz)
			DIV_32	5	Divide by 32 (500 kHz)
			DIV_64	6	Divide by 64 (250 kHz)
			DIV_128	7	Divide by 128 (125 kHz)

6.10.5.25 DECODER

Address offset: 0x510

Configuration of the decoder

Bit number	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
ID																									B						A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LOAD			How a sequence is read from RAM and spread to the compare register
			Common	0	1st half word (16-bit) used in all PWM channels 0..3
			Grouped	1	1st half word (16-bit) used in channel 0..1; 2nd word in channel 2..3
			Individual	2	1st half word (16-bit) in ch.0; 2nd in ch.1; ...; 4th in ch.3
			WaveForm	3	1st half word (16-bit) in ch.0; 2nd in ch.1; ...; 4th in COUNTERTOP
B	RW	MODE			Selects source for advancing the active sequence
			RefreshCount	0	SEQ[n].REFRESH is used to determine loading internal compare registers
			NextStep	1	NEXTSTEP task causes a new value to be loaded to internal compare registers

6.10.5.26 LOOP

Address offset: 0x514

Number of playbacks of a loop

Bit number	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
ID	A A A A A A A A A A A A A A A A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CNT			Number of playbacks of pattern cycles
			Disabled	0	Looping disabled (stop at the end of the sequence)

6.10.5.27 SEQ[n].PTR (n=0..1)

Address offset: 0x520 + (n × 0x20)

Beginning address in RAM of this sequence

Bit number	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
ID	A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			Beginning address in RAM of this sequence

> **Note:** See the memory chapter for details about which memories are available for EasyDMA.

6.10.5.28 SEQ[n].CNT (n=0..1)

Address offset: 0x524 + (n × 0x20)

Number of values (duty cycles) in this sequence

Bit number	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
ID	A A A A A A A A A A A A A A A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CNT			Number of values (duty cycles) in this sequence
			Disabled	0	Sequence is disabled, and shall not be started as it is empty

6.10.5.29 SEQ[n].REFRESH (n=0..1)

Address offset: 0x528 + (n × 0x20)

Number of additional PWM periods between samples loaded into compare register

Bit number	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
ID	A A A A A A A A A A A A A A A A A A A A A A A A
Reset 0x00000001	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1**

ID	Acce	Field	Value ID	Value	Description
A	RW	CNT			Number of additional PWM periods between samples loaded into compare register (load every REFRESH.CNT+1 PWM periods)
			Continuous	0	Update every PWM period

6.10.5.30 SEQ[n].ENDDELAY (n=0..1)

Address offset: 0x52C + (n × 0x20)

Time added after the sequence

Bit number	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
ID									A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CNT			Time added after the sequence in PWM periods

6.10.5.31 PSEL.OUT[n] (n=0..3)

Address offset: 0x560 + (n × 0x4)

Output pin select for PWM channel n

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.11 RTC — Real-time counter

The real-time counter (RTC) module provides a generic, low power timer on the low frequency clock source (LFCLK).

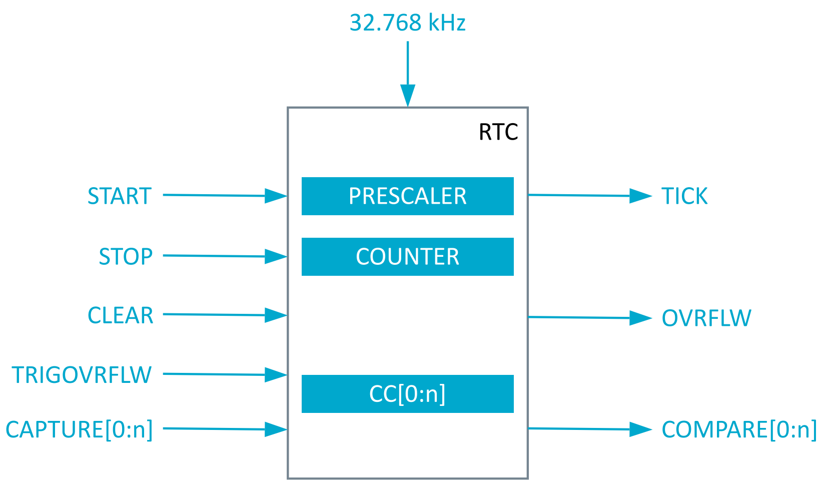

Figure 50: RTC block diagram

The RTC module features a 24-bit COUNTER, a 12-bit (1/X) prescaler, capture/compare registers, and a tick event generator for low power, tickless RTOS implementation.

6.11.1 Clock source

The RTC will run off the LFCLK.

When started, the RTC will automatically request the LFCLK source with RC oscillator if the LFCLK is not already running.

See CLOCK — Clock control on page 65 for more information about clock sources.

6.11.2 Resolution versus overflow and the prescaler

The relationship between the prescaler, counter resolution and overflow is summarized in a table.

Prescaler	Counter resolution	Overflow
0	30.517 µs	512 seconds
2^8-1	7812.5 µs	131072 seconds
2^{12}-1	125 ms	582.542 hours

Table 61: RTC resolution versus overflow

Counter increment frequency is given by the following equation:

$$f_{RTC}\ [kHz] = 32.768\ /\ (PRESCALER + 1)$$

The PRESCALER register is read/write when the RTC is stopped. Once the RTC is started, the prescaler register is read-only and thus writing to it when the RTC is started has no effect.

The prescaler is restarted on tasks START, CLEAR and TRIGOVRFLW. That is, the prescaler value is latched to an internal register (<<PRESC>>) on these tasks.

Examples:

1. Desired COUNTER frequency 100 Hz (10 ms counter period)

   PRESCALER = round(32.768 kHz / 100 Hz) - 1 = 327

   f_{RTC} = 99.9 Hz

   10009.576 µs counter period
2. Desired COUNTER frequency 8 Hz (125 ms counter period)

   PRESCALER = round(32.768 kHz / 8 Hz) – 1 = 4095

   f_{RTC} = 8 Hz

   125 ms counter period

6.11.3 Counter register

The counter increments on LFCLK when the internal PRESCALER register (<<PRESC>>) is 0x00. <<PRESC>> is reloaded from the PRESCALER register. If enabled, the TICK event occurs on each increment of the COUNTER. The TICK event is disabled by default.

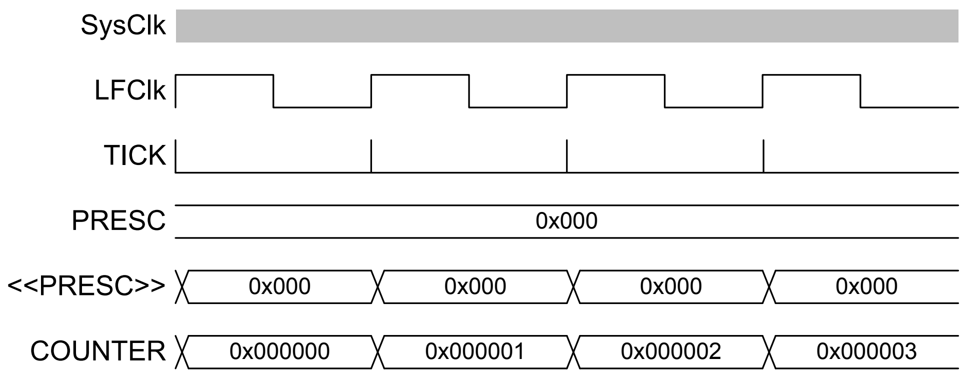

Figure 51: Timing diagram - COUNTER_PRESCALER_0

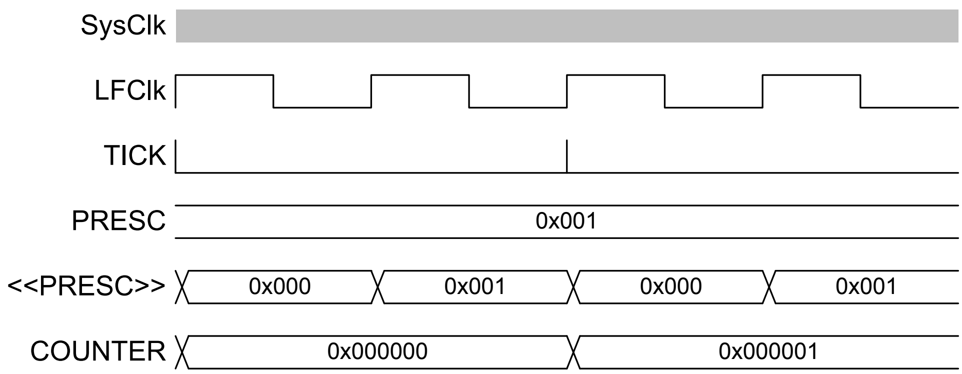

Figure 52: Timing diagram - COUNTER_PRESCALER_1

6.11.4 Overflow

An OVRFLW event is generated on COUNTER register overflow (overflowing from 0xFFFFFF to 0).

The TRIGOVRFLW task will then set the COUNTER value to 0xFFFFF0, to allow software test of the overflow condition.

> **Note:** The OVRFLW event is disabled by default.

6.11.5 Tick event

The TICK event enables low power tick-less RTOS implementation, as it optionally provides a regular interrupt source for an RTOS without the need to use the ARM® SysTick feature.

Using the TICK event, rather than the SysTick, allows the CPU to be powered down while still keeping RTOS scheduling active.

> **Note:** The TICK event is disabled by default.

6.11.6 Event control

To optimize RTC power consumption, events in the RTC can be individually disabled to prevent PCLK16M and HFCLK from being requested when those events are triggered. This is managed using the EVTEN register.

For example, if the TICK event is not required for an application, it should be disabled, since its frequent occurrences may increase power consumption when HFCLK otherwise could be powered down for long periods of time.

This means that the RTC implements a slightly different task and event system compared to the standard system described in Peripheral interface on page 14. The RTC task and event system is illustrated in the figure below.

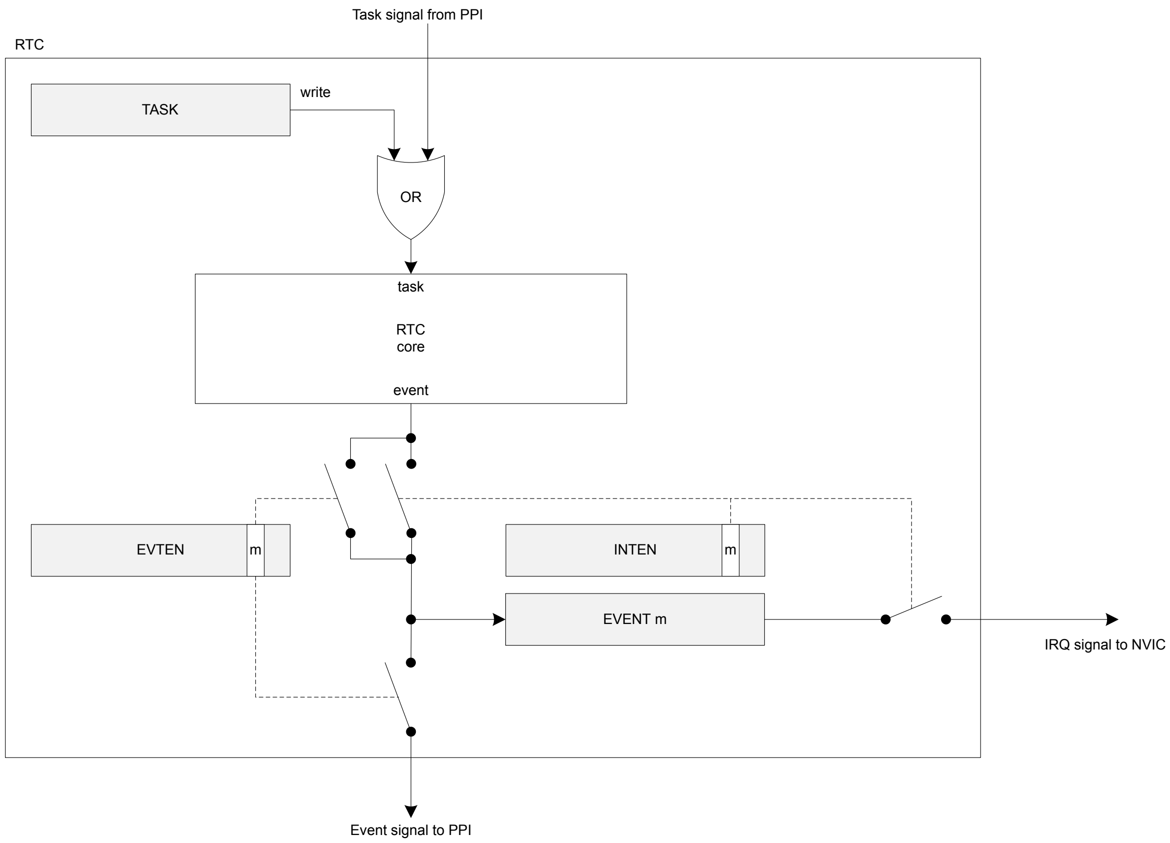

Figure 53: Tasks, events and interrupts in the RTC

6.11.7 Compare

The RTC implements one COMPARE event for every available capture/compare register.

When the COUNTER is incremented and then becomes equal to the value specified in the capture compare register CC[n], the corresponding compare event COMPARE[n] is generated.

When setting a compare register, the following behavior of the RTC COMPARE event should be noted:

- If a CC register value is 0 when a CLEAR task is set, this will not trigger a COMPARE event.

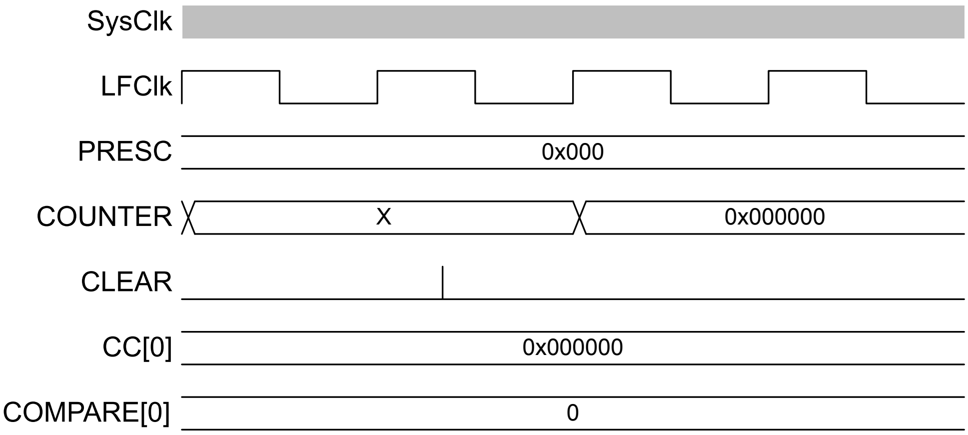

Figure 54: Timing diagram - COMPARE_CLEAR

- If a CC register is N and the COUNTER value is N when the START task is set, this will not trigger a COMPARE event.

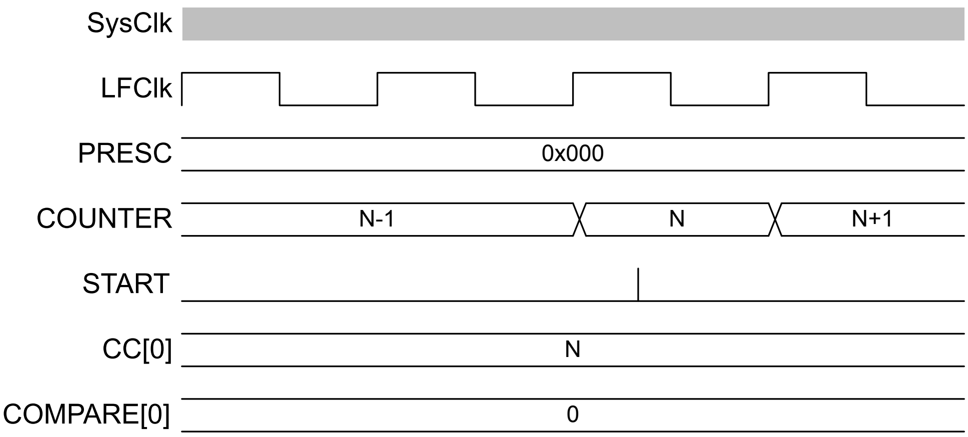

Figure 55: Timing diagram - COMPARE_START

- A COMPARE event occurs when a CC register is N and the COUNTER value transitions from N-1 to N.

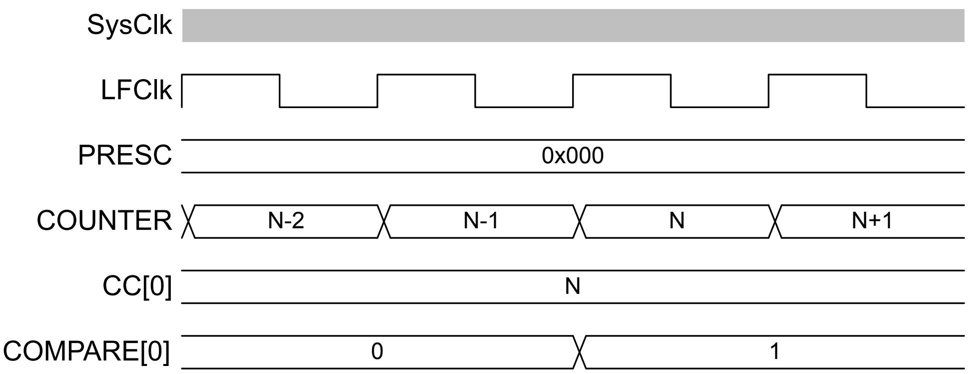

Figure 56: Timing diagram - COMPARE

- If the COUNTER is N, writing N+2 to a CC register is guaranteed to trigger a COMPARE event at N+2.

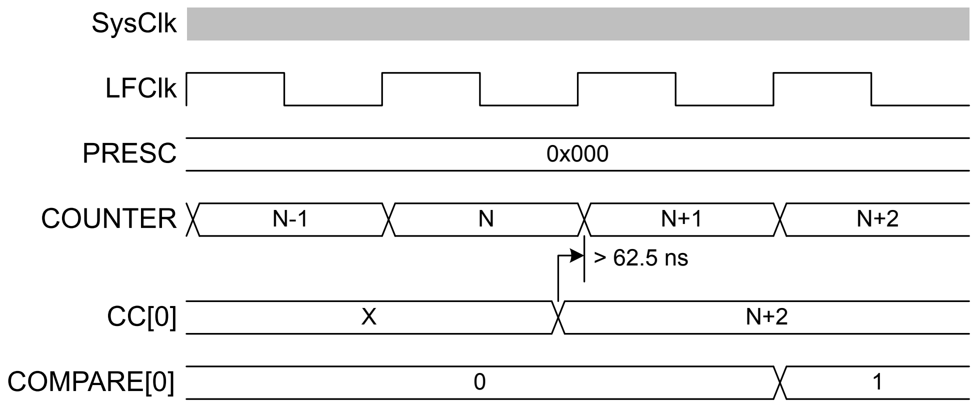

Figure 57: Timing diagram - COMPARE_N+2

- If the COUNTER is N, writing N or N+1 to a CC register may not trigger a COMPARE event.

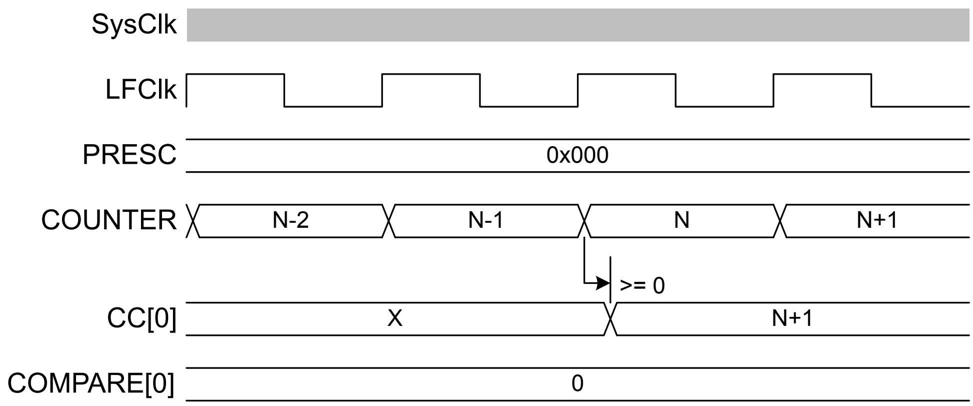

Figure 58: Timing diagram - COMPARE_N+1

- If the COUNTER is N and the current CC register value is N+1 or N+2 when a new CC value is written, a match may trigger on the previous CC value before the new value takes effect. If the current CC value greater than N+2 when the new value is written, there will be no event due to the old value.

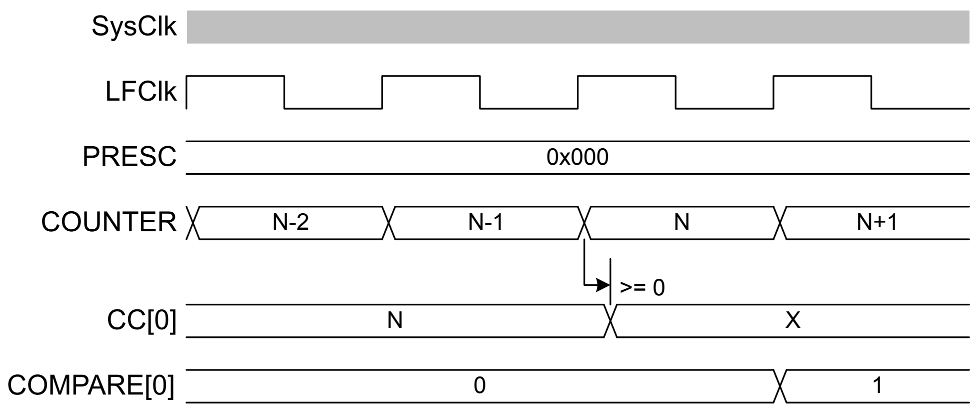

Figure 59: Timing diagram - COMPARE_N-1

6.11.8 Task and event jitter/delay

Jitter or delay in the RTC is due to the peripheral clock being a low frequency clock (LFCLK) which is not synchronous to the faster PCLK16M.

Registers in the peripheral interface, part of the PCLK16M domain, have a set of mirrored registers in the LFCLK domain. For example, the COUNTER value accessible from the CPU is in the PCLK16M domain and is latched on read from an internal COUNTER register in the LFCLK domain. The COUNTER register is modified each time the RTC ticks. The registers are synchronised between the two clock domains (PCLK16M and LFCLK).

1. CLEAR and STOP (and TRIGOVRFLW; not shown) will be delayed as long as it takes for the peripheral to clock a falling edge and rising of the LFCLK. This is between 15.2585 µs and 45.7755 µs – rounded to 15 µs and 46 µs for the remainder of the section.

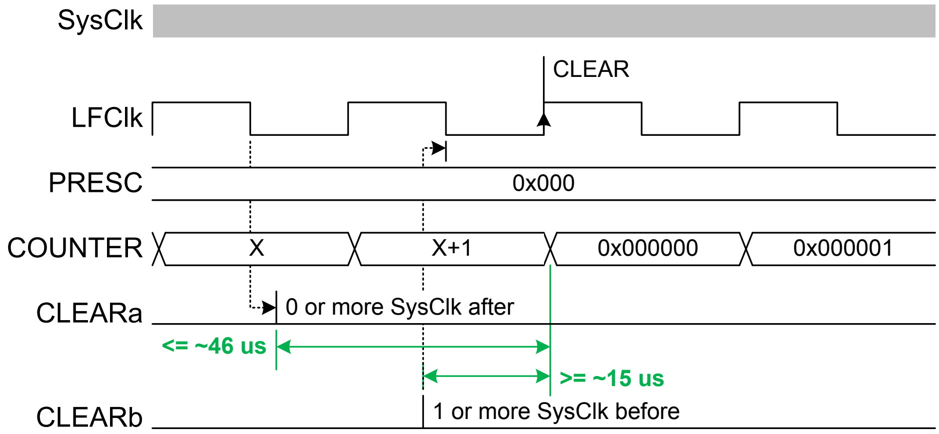

Figure 60: Timing diagram - DELAY_CLEAR

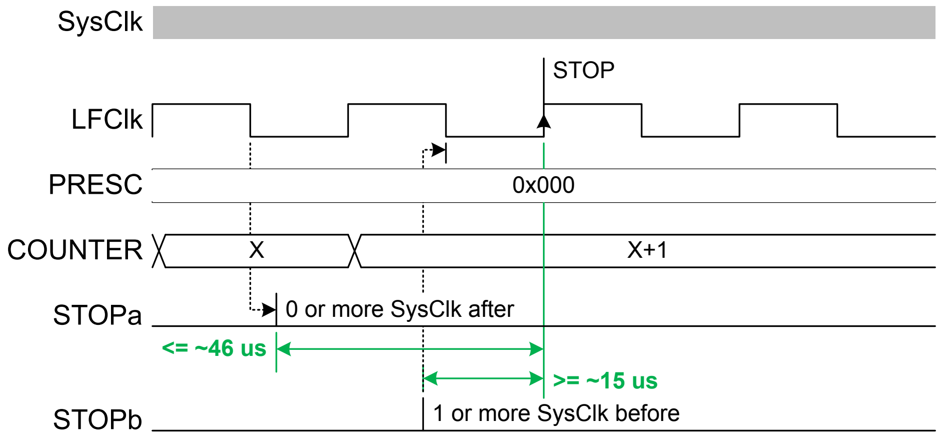

Figure 61: Timing diagram - DELAY_STOP

2. The START task will start the RTC. Assuming that the LFCLK was previously running and stable, the first increment of COUNTER (and instance of TICK event) will be typically after 30.5 µs +/-15 µs. In some cases, in particular if the RTC is started before the LFCLK is running, that timing can be up to ~250 µs. The software should therefore wait for the first TICK if it has to make sure the RTC is running. Sending a TRIGOVRFLW task sets the COUNTER to a value close to overflow. However, since the update of COUNTER relies on a stable LFCLK, sending this task while LFCLK is not running will start LFCLK, but the update will then be delayed by the same amount of time of up to ~250 µs. The figures show the smallest and largest delays on the START task, appearing as a +/-15 µs jitter on the first COUNTER increment.

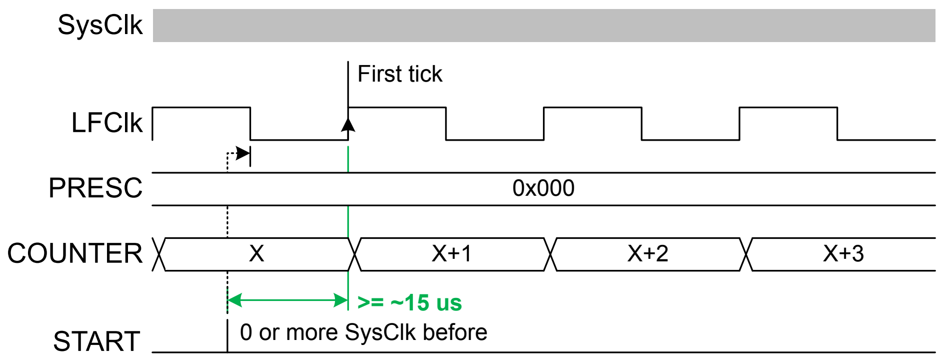

Figure 62: Timing diagram - JITTER_START-

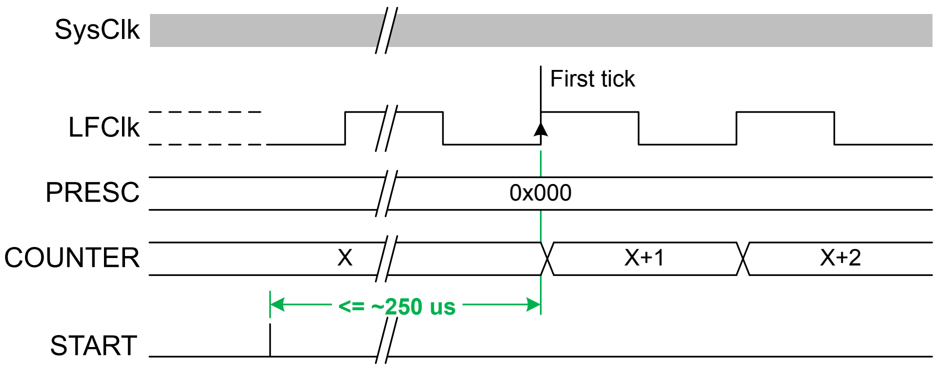

Figure 63: Timing diagram - JITTER_START+

Tables below summarize the jitter introduced on tasks and events.

Task	Delay
CLEAR, START, STOP, TRIGOVRFLOW	+15 to 46 µs

Table 62: RTC jitter magnitudes on tasks

Operation/Function	Jitter
START to COUNTER increment	+/- 15 µs
COMPARE to COMPARE [8]	+/- 62.5 ns

Table 63: RTC jitter magnitudes on events

6.11.9 Reading the counter register

To read the COUNTER register, the internal <<COUNTER>> value is sampled.

To ensure that the <<COUNTER>> is safely sampled (considering that an LFCLK transition may occur during a read), the CPU and core memory bus are halted for three cycles by lowering the core PREADY signal. The read takes the CPU two cycles in addition, resulting in the COUNTER register read taking fixed five PCLK16M clock cycles.

6.11.10 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50014000 0x40014000	RTC	RTC0 : S RTC0 : NS	US	NA	Real time counter 0	
0x50015000 0x40015000	RTC	RTC1 : S RTC1 : NS	US	NA	Real time counter 1	

Table 64: Instances

Register	Offset	Security	Description
TASKS_START	0x000		Start RTC counter
TASKS_STOP	0x004		Stop RTC counter
TASKS_CLEAR	0x008		Clear RTC counter

[8] Assumes RTC runs continuously between these events.

Note: 32.768 kHz clock jitter is additional to the numbers provided above.

Register	Offset	Security	Description
TASKS_TRIGOVRFLW	0x00C		Set counter to 0xFFFFF0
SUBSCRIBE_START	0x080		Subscribe configuration for task START
SUBSCRIBE_STOP	0x084		Subscribe configuration for task STOP
SUBSCRIBE_CLEAR	0x088		Subscribe configuration for task CLEAR
SUBSCRIBE_TRIGOVRFLW	0x08C		Subscribe configuration for task TRIGOVRFLW
EVENTS_TICK	0x100		Event on counter increment
EVENTS_OVRFLW	0x104		Event on counter overflow
EVENTS_COMPARE[0]	0x140		Compare event on CC[0] match
EVENTS_COMPARE[1]	0x144		Compare event on CC[1] match
EVENTS_COMPARE[2]	0x148		Compare event on CC[2] match
EVENTS_COMPARE[3]	0x14C		Compare event on CC[3] match
PUBLISH_TICK	0x180		Publish configuration for event TICK
PUBLISH_OVRFLW	0x184		Publish configuration for event OVRFLW
PUBLISH_COMPARE[0]	0x1C0		Publish configuration for event COMPARE[0]
PUBLISH_COMPARE[1]	0x1C4		Publish configuration for event COMPARE[1]
PUBLISH_COMPARE[2]	0x1C8		Publish configuration for event COMPARE[2]
PUBLISH_COMPARE[3]	0x1CC		Publish configuration for event COMPARE[3]
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
EVTEN	0x340		Enable or disable event routing
EVTENSET	0x344		Enable event routing
EVTENCLR	0x348		Disable event routing
COUNTER	0x504		Current counter value
PRESCALER	0x508		12-bit prescaler for counter frequency (32768/(PRESCALER+1)). Must be written when RTC is stopped.
CC[n]	0x540		Compare register n

Table 65: Register overview

6.11.10.1 TASKS_START

Address offset: 0x000

Start RTC counter

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_START			Start RTC counter
			Trigger	1	Trigger task

6.11.10.2 TASKS_STOP

Address offset: 0x004

Stop RTC counter

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_STOP			Stop RTC counter
			Trigger	1	Trigger task

6.11.10.3 TASKS_CLEAR

Address offset: 0x008

Clear RTC counter

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_CLEAR			Clear RTC counter
			Trigger	1	Trigger task

6.11.10.4 TASKS_TRIGOVRFLW

Address offset: 0x00C

Set counter to 0xFFFFF0

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_TRIGOVRFLW			Set counter to 0xFFFFF0
			Trigger	1	Trigger task

6.11.10.5 SUBSCRIBE_START

Address offset: 0x080

Subscribe configuration for task START

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task START will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.11.10.6 SUBSCRIBE_STOP

Address offset: 0x084

Subscribe configuration for task STOP

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STOP will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.11.10.7 SUBSCRIBE_CLEAR

Address offset: 0x088

Subscribe configuration for task CLEAR

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task CLEAR will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.11.10.8 SUBSCRIBE_TRIGOVRFLW

Address offset: 0x08C

Subscribe configuration for task TRIGOVRFLW

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task TRIGOVRFLW will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.11.10.9 EVENTS_TICK

Address offset: 0x100

Event on counter increment

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_TICK			Event on counter increment
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.11.10.10 EVENTS_OVRFLW

Address offset: 0x104

Event on counter overflow

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_OVRFLW			Event on counter overflow
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.11.10.11 EVENTS_COMPARE[n] (n=0..3)

Address offset: 0x140 + (n × 0x4)

Compare event on CC[n] match

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_COMPARE			Compare event on CC[n] match
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.11.10.12 PUBLISH_TICK

Address offset: 0x180

Publish configuration for event TICK

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event TICK will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.11.10.13 PUBLISH_OVRFLW

Address offset: 0x184

Publish configuration for event OVRFLW

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event OVRFLW will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.11.10.14 PUBLISH_COMPARE[n] (n=0..3)

Address offset: 0x1C0 + (n × 0x4)

Publish configuration for event COMPARE[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event COMPARE[n] will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.11.10.15 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID													F	E	D	C															B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	TICK			Write '1' to enable interrupt for event TICK
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	OVRFLW			Write '1' to enable interrupt for event OVRFLW
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C-F	RW	COMPARE[i] (i=0..3)			Write '1' to enable interrupt for event COMPARE[i]
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.11.10.16 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID													F	E	D	C															B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	TICK			Write '1' to disable interrupt for event TICK
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	OVRFLW			Write '1' to disable interrupt for event OVRFLW
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C-F	RW	COMPARE[i] (i=0..3)			Write '1' to disable interrupt for event COMPARE[i]
			Clear	1	Disable

Bit number	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
ID													F	E	D	C															B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.11.10.17 EVTEN

Address offset: 0x340

Enable or disable event routing

Bit number	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
ID													F	E	D	C															B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	TICK			Enable or disable event routing for event TICK
			Disabled	0	Disable
			Enabled	1	Disable
B	RW	OVRFLW			Enable or disable event routing for event OVRFLW
			Disabled	0	Disable
			Enabled	1	Disable
C-F	RW	COMPARE[i] (i=0..3)			Enable or disable event routing for event COMPARE[i]
			Disabled	0	Disable
			Enabled	1	Disable

6.11.10.18 EVTENSET

Address offset: 0x344

Enable event routing

Bit number	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
ID													F	E	D	C															B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	TICK			Write '1' to enable event routing for event TICK
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
			Set	1	Enable
B	RW	OVRFLW			Write '1' to enable event routing for event OVRFLW
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
			Set	1	Enable
C-F	RW	COMPARE[i] (i=0..3)			Write '1' to enable event routing for event COMPARE[i]
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
			Set	1	Enable

6.11.10.19 EVTENCLR

Address offset: 0x348

Disable event routing

Bit number	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
ID													F	E	D	C															B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	TICK			Write '1' to disable event routing for event TICK
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
			Clear	1	Disable
B	RW	OVRFLW			Write '1' to disable event routing for event OVRFLW
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
			Clear	1	Disable
C-F	RW	COMPARE[i] (i=0..3)			Write '1' to disable event routing for event COMPARE[i]
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
			Clear	1	Disable

6.11.10.20 COUNTER

Address offset: 0x504

Current counter value

Bit number	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
ID									A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	COUNTER			Counter value

6.11.10.21 PRESCALER

Address offset: 0x508

12-bit prescaler for counter frequency (32768/(PRESCALER+1)). Must be written when RTC is stopped.

Bit number	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
ID																					A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PRESCALER			Prescaler value

6.11.10.22 CC[n] (n=0..3)

Address offset: 0x540 + (n × 0x4)

Compare register n

Bit number	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
ID									A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	COMPARE			Compare value

6.11.11 Electrical specification

6.12 SAADC — Successive approximation analog-to-digital converter

The SAADC is a differential successive approximation register (SAR) analog-to-digital converter.

Listed here are the main features of SAADC:

- 8/10/12-bit resolution, 14-bit resolution with oversampling
- Up to eight input channels
  - One channel per single-ended input and two channels per differential input
  - Scan mode can be configured with both single-ended channels and differential channels.
- Full scale input range (0 to VDD_GPIO)
- Sampling triggered via a task from software or a PPI channel for full flexibility on sample frequency source from low power 32.768kHz RTC or more accurate 1/16MHz Timers
- One-shot conversion mode to sample a single channel
- Scan mode to sample a series of channels in sequence. Sample delay between channels is $t_{ack} + t_{conv}$ which may vary between channels according to user configuration of t_{ack}.
- Support for direct sample transfer to RAM using EasyDMA
- Interrupts on single sample and full buffer events
- Samples stored as 16-bit 2's complement values for differential and single-ended sampling
- Continuous sampling without the need of an external timer
- Internal resistor string
- Limit checking on the fly

6.12.1 Overview

The ADC supports up to eight external analog input channels. It can be operated in a one-shot mode with sampling under software control, or a continuous conversion mode with a programmable sampling rate.

The analog inputs can be configured as eight single-ended inputs, four differential inputs or a combination of these. Each channel can be configured to select `AIN0` to `AIN7` pins, or the VDD_GPIO pin. Channels can be sampled individually in one-shot or continuous sampling modes, or, using scan mode, multiple channels can be sampled in sequence. Channels can also be oversampled to improve noise performance.

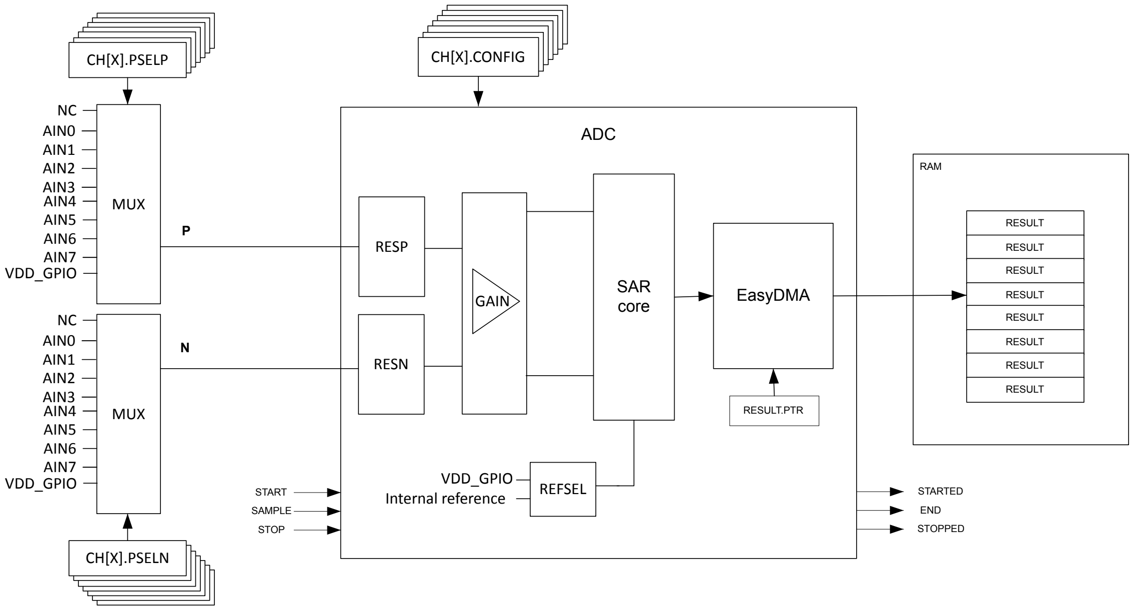

Figure 64: Simplified ADC block diagram

Internally, the ADC is always a differential analog-to-digital converter, but by default it is configured with single-ended input in the MODE field of the CH[n].CONFIG register. In single-ended mode, the negative input will be shorted to ground internally.

The assumption in single-ended mode is that the internal ground of the ADC is the same as the external ground that the measured voltage is referred to. The ADC is thus sensitive to ground bounce on the PCB in single-ended mode. If this is a concern we recommend using differential measurement.

6.12.2 Digital output

The output result of the ADC depends on the settings in the CH[n].CONFIG and RESOLUTION registers as follows:

```
RESULT = [V(P) – V(N) ] * GAIN/REFERENCE * 2^(RESOLUTION - m)
```

where

V(P)

is the voltage at input P

V(N)

is the voltage at input N

GAIN

is the selected gain setting

REFERENCE

is the selected reference voltage

and m=0 if CONFIG.MODE=SE, or m=1 if CONFIG.MODE=Diff.

The result generated by the ADC will deviate from the expected due DC errors like offset, gain, differential non-linearity (DNL), and integral non-linearity (INL). See Electrical specification for details on these parameters. The result can also vary due to AC errors like non-linearities in the GAIN block, settling errors due to high source impedance and sampling jitter. For battery measurement the DC errors are most noticeable.

The ADC has a wide selection of gains controlled in the GAIN field of the CH[n].CONFIG register. If CH[n].CONFIG.REFSEL=0, the input range of the ADC core is nominally ±0.6 V differential and the input must be scaled accordingly.

The ADC has a temperature dependent offset. If the ADC is to operate over a large temperature range, we recommend running CALIBRATEOFFSET at regular intervals. The CALIBRATEDONE event will be fired when the calibration has been completed. Note that the DONE and RESULTDONE events will also be generated.

6.12.3 Analog inputs and channels

Up to eight analog input channels, CH[n](n=0..7), can be configured.

Any one of the available channels can be enabled for the ADC to operate in one-shot mode. If more than one CH[n] is configured, the ADC enters scan mode.

An analog input is selected as a positive converter input if CH[n].PSELP is set, setting CH[n].PSELP also enables the particular channel.

An analog input is selected as a negative converter input if CH[n].PSELN is set. The CH[n].PSELN register will have no effect unless differential mode is enabled, see MODE field in CH[n].CONFIG register.

If more than one of the CH[n].PSELP registers is set, the device enters scan mode. Input selections in scan mode are controlled by the CH[n].PSELP and CH[n].PSELN registers, where CH[n].PSELN is only used if the particular scan channel is specified as differential, see MODE field in CH[n].CONFIG register.

6.12.4 Operation modes

The ADC input configuration supports one-shot mode, continuous mode and scan mode.

> **Note:** Scan mode and oversampling cannot be combined.

The ADC indicates a single ongoing conversion via the register STATUS on page 221. During scan mode, oversampling, or continuous modes, more than a single conversion take place in the ADC. As consequence, the value reflected in STATUS register will toggle at the end of each single conversion.

6.12.4.1 One-shot mode

One-shot operation is configured by enabling only one of the available channels defined by CH[n].PSELP, CH[n].PSELN, and CH[n].CONFIG registers.

Upon a SAMPLE task, the ADC starts to sample the input voltage. The CH[n].CONFIG.TACQ controls the acquisition time.

A DONE event signals that one sample has been taken.

In this mode, the RESULTDONE event has the same meaning as DONE when no oversampling takes place. Note that both events may occur before the actual value has been transferred into RAM by EasyDMA. For more information, see EasyDMA on page 203.

6.12.4.2 Continuous mode

Continuous sampling can be achieved by using the internal timer in the ADC, or triggering the SAMPLE task from one of the general purpose timers through the PPI system.

Care shall be taken to ensure that the sample rate fulfils the following criteria, depending on how many channels are active:

$$f_{SAMPLE} < 1/[t_{ACQ} + t_{conv}]$$

The SAMPLERATE register can be used as a local timer instead of triggering individual SAMPLE tasks. When SAMPLERATE.MODE is set to Timers, it is sufficient to trigger SAMPLE task only once in order to start the SAADC and triggering the STOP task will stop sampling. The SAMPLERATE.CC field controls the sample rate.

The SAMPLERATE timer mode cannot be combined with SCAN mode, and only one channel can be enabled in this mode.

A DONE event signals that one sample has been taken.

In this mode, the RESULTDONE event has the same meaning as DONE when no oversampling takes place. Note that both events may occur before the actual value has been transferred into RAM by EasyDMA.

6.12.4.3 Oversampling

An accumulator in the ADC can be used to average noise on the analog input. In general, oversampling improves the signal-to-noise ratio (SNR). Oversampling, however, does not improve the integral non-linearity (INL), or differential non-linearity (DNL).

Oversampling and scan should not be combined, since oversampling and scan will average over input channels.

The accumulator is controlled in the OVERSAMPLE register. The SAMPLE task must be set $2^{OVERSAMPLE}$ number of times before the result is written to RAM. This can be achieved by:

- Configuring a fixed sampling rate using the local timer or a general purpose timer and the PPI system to trigger a SAMPLE task
- Triggering SAMPLE $2^{OVERSAMPLE}$ times from software
- Enabling BURST mode

CH[n].CONFIG.BURST can be enabled to avoid setting SAMPLE task $2^{OVERSAMPLE}$ times. With BURST = 1 the ADC will sample the input $2^{OVERSAMPLE}$ times as fast as it can (actual timing: $<(t_{ACQ}+t_{CONV})\times 2^{OVERSAMPLE}$). Thus, for the user it will just appear like the conversion took a bit longer time, but other than that, it is similar to one-shot mode.

A DONE event signals that one sample has been taken.

In this mode, the RESULTDONE event signals that enough conversions have taken place for an oversampled result to get transferred into RAM. Note that both events may occur before the actual value has been transferred into RAM by EasyDMA.

6.12.4.4 Scan mode

A channel is considered enabled if CH[n].PSELP is set. If more than one channel, CH[n], is enabled, the ADC enters scan mode.

In scan mode, one SAMPLE task will trigger one conversion per enabled channel. The time it takes to sample all channels is:

```
Total time < Sum(CH[x].tACQ+tCONV), x=0..enabled channels
```

A DONE event signals that one sample has been taken.

In this mode, the RESULTDONE event signals has the same meaning as DONE when no oversampling takes place. Note that both events may occur before the actual values have been transferred into RAM by EasyDMA.

Example of RAM placement (even RESULT.MAXCNT), channels 1, 2 and 5 enabled on page 203 provides an example of results placement in Data RAM, with an even RESULT.MAXCNT. In this example, channels 1, 2 and 5 are enabled, all others are disabled.

	31 … 16	15 … 0
RESULT.PTR	CH[2] 1st result	CH[1] 1st result
RESULT.PTR + 4	CH[1] 2nd result	CH[5] 1st result
RESULT.PTR + 8	CH[5] 2nd result	CH[2] 2nd result
	(...)	
RESULT.PTR + 2*(RESULT.MAXCNT – 2)	CH[5] last result	CH[2] last result

Figure 65: Example of RAM placement (even RESULT.MAXCNT), channels 1, 2 and 5 enabled

Example of RAM placement (odd RESULT.MAXCNT), channels 1, 2 and 5 enabled on page 203 provides an example of results placement in Data RAM, with an odd RESULT.MAXCNT. In this example, channels 1, 2 and 5 are enabled, all others are disabled. The last 32-bit word is populated only with one 16-bit result.

	31 … 16	15 … 0
RESULT.PTR	CH[2] 1st result	CH[1] 1st result
RESULT.PTR + 4	CH[1] 2nd result	CH[5] 1st result
RESULT.PTR + 8	CH[5] 2nd result	CH[2] 2nd result
	(...)	
RESULT.PTR + 2*(RESULT.MAXCNT – 1)		CH[5] last result

Figure 66: Example of RAM placement (odd RESULT.MAXCNT), channels 1, 2 and 5 enabled

6.12.5 EasyDMA

After configuring RESULT.PTR and RESULT.MAXCNT, the ADC resources are started by triggering the START task. The ADC is using EasyDMA to store results in a Result buffer in RAM.

The Result buffer is located at the address specified in the RESULT.PTR register. The RESULT.PTR register is double-buffered and it can be updated and prepared for the next START task immediately after the STARTED event is generated. The size of the Result buffer is specified in the RESULT.MAXCNT register and the ADC will generate an END event when it has filled up the Result buffer, see ADC on page 204. Results are stored in little-endian byte order in Data RAM. Every sample will be sign extended to 16 bit before stored in the Result buffer.

The ADC is stopped by triggering the STOP task. The STOP task will terminate an ongoing sampling. The ADC will generate a STOPPED event when it has stopped. If the ADC is already stopped when the STOP task is triggered, the STOPPED event will still be generated.

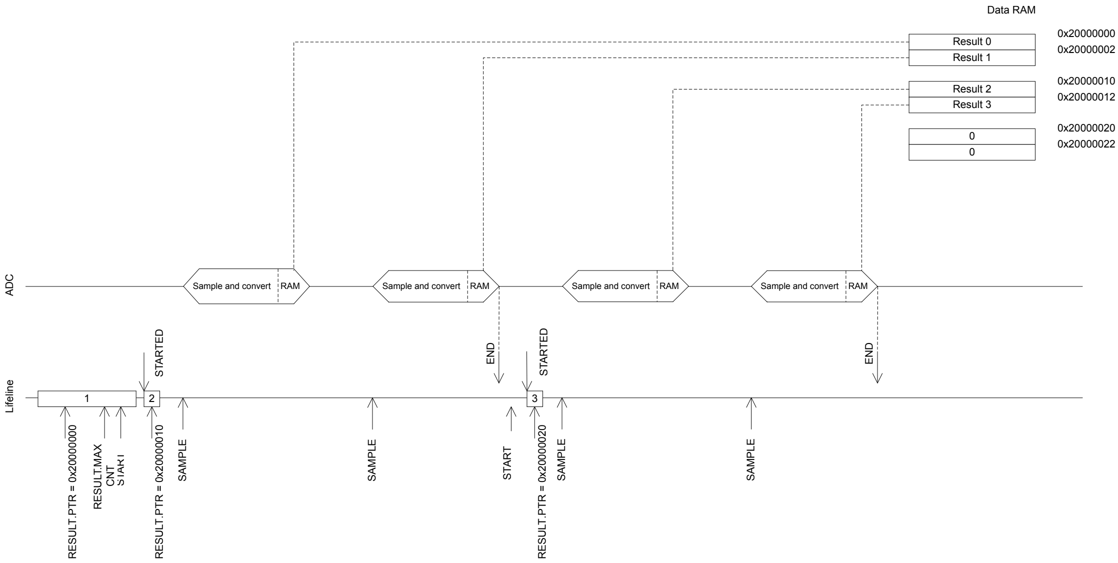

Figure 67: ADC

If the RESULT.PTR is not pointing to the Data RAM region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 20 for more information about the different memory regions.

The EasyDMA will have finished accessing the RAM when the END or STOPPED event has been generated.

The RESULT.AMOUNT register can be read following an END event or a STOPPED event to see how many results have been transferred to the Result buffer in RAM since the START task was triggered.

In scan mode, SAMPLE tasks can be triggered once the START task is triggered. The END event is generated when the number of samples transferred to memory reaches the value specified by RESULT.MAXCNT. After an END event, the START task needs to be triggered again before new samples can be taken. Also make sure that the size of the Result buffer is large enough to have space for minimum one result from each of the enabled channels, by specifying RESULT.MAXCNT >= number of channels enabled. For more information about the scan mode, see Scan mode on page 202.

6.12.6 Resistor ladder

The ADC has an internal resistor string for positive and negative input.

See Resistor ladder for positive input (negative input is equivalent, using RESN instead of RESP) on page 205. The resistors are controlled in the CH[n].CONFIG.RESP and CH[n].CONFIG.RESN registers.

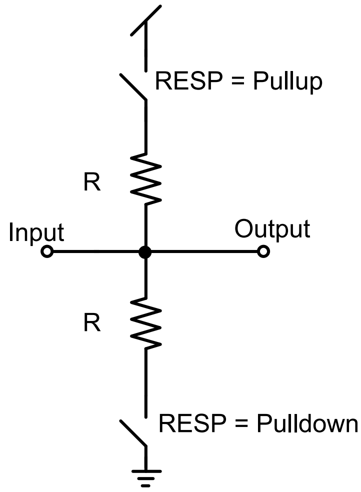

Figure 68: Resistor ladder for positive input (negative input is equivalent, using RESN instead of RESP)

6.12.7 Reference

The ADC can use two different references, controlled in the REFSEL field of the CH[n].CONFIG register.

These are:

- Internal reference
- VDD_GPIO as reference

The internal reference results in an input range of ±0.6 V on the ADC core. VDD_GPIO as reference results in an input range of ±VDD_GPIO/4 on the ADC core. The gain block can be used to change the effective input range of the ADC.

```
Input range = (+- 0.6 V or +-VDD_GPIO/4)/Gain
```

For example, choosing VDD_GPIO as reference, single ended input (grounded negative input), and a gain of 1/4 the input range will be:

```
Input range = (VDD_GPIO/4)/(1/4) = VDD_GPIO
```

With internal reference, single ended input (grounded negative input), and a gain of 1/6 the input range will be:

```
Input range = (0.6 V)/(1/6) = 3.6 V
```

The AIN0-AIN7 inputs cannot exceed VDD_GPIO, or be lower than VSS.

6.12.8 Acquisition time

To sample the input voltage, the ADC connects a capacitor to the input.

For illustration, see Simplified ADC sample network on page 206. The acquisition time indicates how long the capacitor is connected, see TACQ field in CH[n].CONFIG register. The required acquisition time depends on the source (R_{source}) resistance. For high source resistance the acquisition time should be increased, see Acquisition time on page 206.

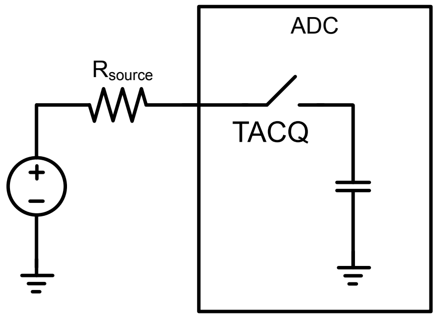

Figure 69: Simplified ADC sample network

TACQ [µs]	Maximum source resistance [kOhm]
3	10
5	40
10	100
15	200
20	400
40	800

Table 66: Acquisition time

6.12.9 Limits event monitoring

A channel can be event monitored by configuring limit register CH[n].LIMIT.

If the conversion result is higher than the defined high limit, or lower than the defined low limit, the appropriate event will get fired.

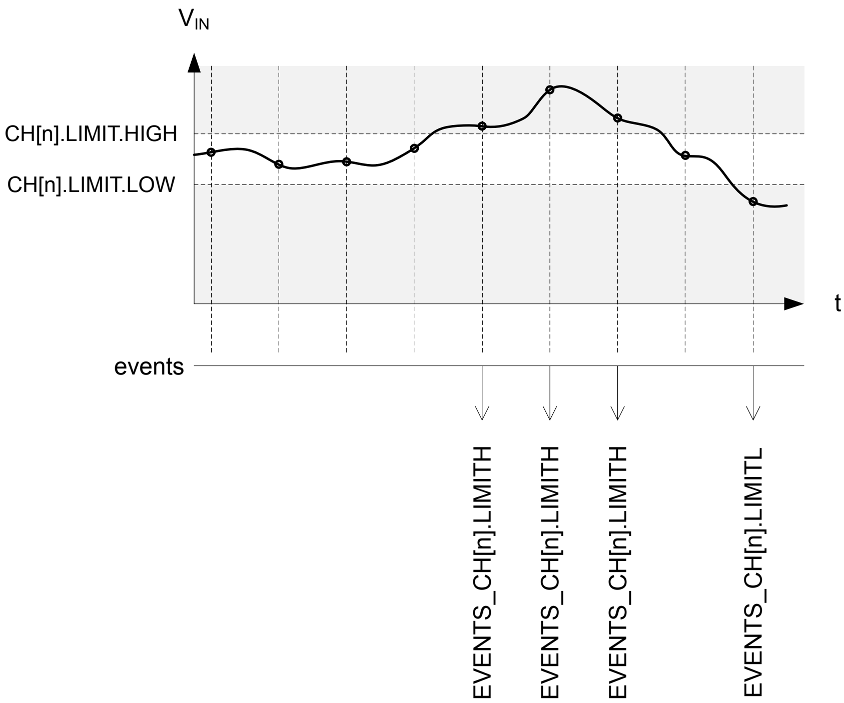

Figure 70: Example of limits monitoring on channel 'n'

Note that when setting the limits, CH[n].LIMIT.HIGH shall always be higher than or equal to CH[n].LIMIT.LOW . In other words, an event can be fired only when the input signal has been sampled outside of the defined limits. It is not possible to fire an event when the input signal is inside a defined range by swapping high and low limits.

The comparison to limits always takes place, there is no need to enable it. If comparison is not required on a channel, the software shall simply ignore the related events. In that situation, the value of the limits registers is irrelevant, so it does not matter if CH[n].LIMIT.LOW is lower than CH[n].LIMIT.HIGH or not.

6.12.10 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x5000E000 0x4000E000	SAADC	SAADC : S SAADC : NS	US	SA	Analog to digital converter	

Table 67: Instances

Register	Offset	Security	Description
TASKS_START	0x000		Start the ADC and prepare the result buffer in RAM
TASKS_SAMPLE	0x004		Take one ADC sample, if scan is enabled all channels are sampled
TASKS_STOP	0x008		Stop the ADC and terminate any on-going conversion
TASKS_CALIBRATEOFFSET	0x00C		Starts offset auto-calibration
SUBSCRIBE_START	0x080		Subscribe configuration for task START
SUBSCRIBE_SAMPLE	0x084		Subscribe configuration for task SAMPLE
SUBSCRIBE_STOP	0x088		Subscribe configuration for task STOP
SUBSCRIBE_CALIBRATEOFF:	0x08C		Subscribe configuration for task CALIBRATEOFFSET
EVENTS_STARTED	0x100		The ADC has started
EVENTS_END	0x104		The ADC has filled up the Result buffer
EVENTS_DONE	0x108		A conversion task has been completed. Depending on the mode, multiple conversions might be needed for a result to be transferred to RAM.
EVENTS_RESULTDONE	0x10C		A result is ready to get transferred to RAM.
EVENTS_CALIBRATEDONE	0x110		Calibration is complete
EVENTS_STOPPED	0x114		The ADC has stopped
EVENTS_CH[0].LIMITH	0x118		Last results is equal or above CH[0].LIMIT.HIGH
EVENTS_CH[0].LIMITL	0x11C		Last results is equal or below CH[0].LIMIT.LOW
EVENTS_CH[1].LIMITH	0x120		Last results is equal or above CH[1].LIMIT.HIGH
EVENTS_CH[1].LIMITL	0x124		Last results is equal or below CH[1].LIMIT.LOW
EVENTS_CH[2].LIMITH	0x128		Last results is equal or above CH[2].LIMIT.HIGH
EVENTS_CH[2].LIMITL	0x12C		Last results is equal or below CH[2].LIMIT.LOW
EVENTS_CH[3].LIMITH	0x130		Last results is equal or above CH[3].LIMIT.HIGH
EVENTS_CH[3].LIMITL	0x134		Last results is equal or below CH[3].LIMIT.LOW
EVENTS_CH[4].LIMITH	0x138		Last results is equal or above CH[4].LIMIT.HIGH
EVENTS_CH[4].LIMITL	0x13C		Last results is equal or below CH[4].LIMIT.LOW
EVENTS_CH[5].LIMITH	0x140		Last results is equal or above CH[5].LIMIT.HIGH
EVENTS_CH[5].LIMITL	0x144		Last results is equal or below CH[5].LIMIT.LOW
EVENTS_CH[6].LIMITH	0x148		Last results is equal or above CH[6].LIMIT.HIGH
EVENTS_CH[6].LIMITL	0x14C		Last results is equal or below CH[6].LIMIT.LOW
EVENTS_CH[7].LIMITH	0x150		Last results is equal or above CH[7].LIMIT.HIGH
EVENTS_CH[7].LIMITL	0x154		Last results is equal or below CH[7].LIMIT.LOW
PUBLISH_STARTED	0x180		Publish configuration for event STARTED
PUBLISH_END	0x184		Publish configuration for event END
PUBLISH_DONE	0x188		Publish configuration for event DONE
PUBLISH_RESULTDONE	0x18C		Publish configuration for event RESULTDONE
PUBLISH_CALIBRATEDONE	0x190		Publish configuration for event CALIBRATEDONE

Register	Offset	Security	Description
PUBLISH_STOPPED	0x194		Publish configuration for event STOPPED
PUBLISH_CH[0].LIMITH	0x198		Publish configuration for event CH[0].LIMITH
PUBLISH_CH[0].LIMITL	0x19C		Publish configuration for event CH[0].LIMITL
PUBLISH_CH[1].LIMITH	0x1A0		Publish configuration for event CH[1].LIMITH
PUBLISH_CH[1].LIMITL	0x1A4		Publish configuration for event CH[1].LIMITL
PUBLISH_CH[2].LIMITH	0x1A8		Publish configuration for event CH[2].LIMITH
PUBLISH_CH[2].LIMITL	0x1AC		Publish configuration for event CH[2].LIMITL
PUBLISH_CH[3].LIMITH	0x1B0		Publish configuration for event CH[3].LIMITH
PUBLISH_CH[3].LIMITL	0x1B4		Publish configuration for event CH[3].LIMITL
PUBLISH_CH[4].LIMITH	0x1B8		Publish configuration for event CH[4].LIMITH
PUBLISH_CH[4].LIMITL	0x1BC		Publish configuration for event CH[4].LIMITL
PUBLISH_CH[5].LIMITH	0x1C0		Publish configuration for event CH[5].LIMITH
PUBLISH_CH[5].LIMITL	0x1C4		Publish configuration for event CH[5].LIMITL
PUBLISH_CH[6].LIMITH	0x1C8		Publish configuration for event CH[6].LIMITH
PUBLISH_CH[6].LIMITL	0x1CC		Publish configuration for event CH[6].LIMITL
PUBLISH_CH[7].LIMITH	0x1D0		Publish configuration for event CH[7].LIMITH
PUBLISH_CH[7].LIMITL	0x1D4		Publish configuration for event CH[7].LIMITL
INTEN	0x300		Enable or disable interrupt
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
STATUS	0x400		Status
ENABLE	0x500		Enable or disable ADC
CH[0].PSELP	0x510		Input positive pin selection for CH[0]
CH[0].PSELN	0x514		Input negative pin selection for CH[0]
CH[0].CONFIG	0x518		Input configuration for CH[0]
CH[0].LIMIT	0x51C		High/low limits for event monitoring a channel
CH[1].PSELP	0x520		Input positive pin selection for CH[1]
CH[1].PSELN	0x524		Input negative pin selection for CH[1]
CH[1].CONFIG	0x528		Input configuration for CH[1]
CH[1].LIMIT	0x52C		High/low limits for event monitoring a channel
CH[2].PSELP	0x530		Input positive pin selection for CH[2]
CH[2].PSELN	0x534		Input negative pin selection for CH[2]
CH[2].CONFIG	0x538		Input configuration for CH[2]
CH[2].LIMIT	0x53C		High/low limits for event monitoring a channel
CH[3].PSELP	0x540		Input positive pin selection for CH[3]
CH[3].PSELN	0x544		Input negative pin selection for CH[3]
CH[3].CONFIG	0x548		Input configuration for CH[3]
CH[3].LIMIT	0x54C		High/low limits for event monitoring a channel
CH[4].PSELP	0x550		Input positive pin selection for CH[4]
CH[4].PSELN	0x554		Input negative pin selection for CH[4]
CH[4].CONFIG	0x558		Input configuration for CH[4]
CH[4].LIMIT	0x55C		High/low limits for event monitoring a channel
CH[5].PSELP	0x560		Input positive pin selection for CH[5]
CH[5].PSELN	0x564		Input negative pin selection for CH[5]
CH[5].CONFIG	0x568		Input configuration for CH[5]
CH[5].LIMIT	0x56C		High/low limits for event monitoring a channel
CH[6].PSELP	0x570		Input positive pin selection for CH[6]
CH[6].PSELN	0x574		Input negative pin selection for CH[6]
CH[6].CONFIG	0x578		Input configuration for CH[6]
CH[6].LIMIT	0x57C		High/low limits for event monitoring a channel
CH[7].PSELP	0x580		Input positive pin selection for CH[7]
CH[7].PSELN	0x584		Input negative pin selection for CH[7]
CH[7].CONFIG	0x588		Input configuration for CH[7]

Register	Offset	Security	Description
CH[7].LIMIT	0x58C		High/low limits for event monitoring a channel
RESOLUTION	0x5F0		Resolution configuration
OVERSAMPLE	0x5F4		Oversampling configuration. OVERSAMPLE should not be combined with SCAN. The RESOLUTION is applied before averaging, thus for high OVERSAMPLE a higher RESOLUTION should be used.
SAMPLERATE	0x5F8		Controls normal or continuous sample rate
RESULT.PTR	0x62C		Data pointer
RESULT.MAXCNT	0x630		Maximum number of buffer words to transfer
RESULT.AMOUNT	0x634		Number of buffer words transferred since last START

Table 68: Register overview

6.12.10.1 TASKS_START

Address offset: 0x000

Start the ADC and prepare the result buffer in RAM

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_START			Start the ADC and prepare the result buffer in RAM
			Trigger	1	Trigger task

6.12.10.2 TASKS_SAMPLE

Address offset: 0x004

Take one ADC sample, if scan is enabled all channels are sampled

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_SAMPLE			Take one ADC sample, if scan is enabled all channels are sampled
			Trigger	1	Trigger task

6.12.10.3 TASKS_STOP

Address offset: 0x008

Stop the ADC and terminate any on-going conversion

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_STOP			Stop the ADC and terminate any on-going conversion
			Trigger	1	Trigger task

6.12.10.4 TASKS_CALIBRATEOFFSET

Address offset: 0x00C

Starts offset auto-calibration

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_CALIBRATEOFFSET			Starts offset auto-calibration
			Trigger	1	Trigger task

6.12.10.5 SUBSCRIBE_START

Address offset: 0x080

Subscribe configuration for task START

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task START will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.12.10.6 SUBSCRIBE_SAMPLE

Address offset: 0x084

Subscribe configuration for task SAMPLE

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task SAMPLE will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.12.10.7 SUBSCRIBE_STOP

Address offset: 0x088

Subscribe configuration for task STOP

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STOP will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.12.10.8 SUBSCRIBE_CALIBRATEOFFSET

Address offset: 0x08C

Subscribe configuration for task CALIBRATEOFFSET

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task CALIBRATEOFFSET will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.12.10.9 EVENTS_STARTED

Address offset: 0x100

The ADC has started

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_STARTED			The ADC has started
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.12.10.10 EVENTS_END

Address offset: 0x104

The ADC has filled up the Result buffer

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_END			The ADC has filled up the Result buffer
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.12.10.11 EVENTS_DONE

Address offset: 0x108

A conversion task has been completed. Depending on the mode, multiple conversions might be needed for a result to be transferred to RAM.

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_DONE			A conversion task has been completed. Depending on the mode, multiple conversions might be needed for a result to be transferred to RAM.
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.12.10.12 EVENTS_RESULTDONE

Address offset: 0x10C

A result is ready to get transferred to RAM.

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_RESULTDONE			A result is ready to get transferred to RAM.
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.12.10.13 EVENTS_CALIBRATEDONE

Address offset: 0x110

Calibration is complete

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_CALIBRATEDONE			Calibration is complete
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.12.10.14 EVENTS_STOPPED

Address offset: 0x114

The ADC has stopped

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_STOPPED			The ADC has stopped
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.12.10.15 EVENTS_CH[n].LIMITH (n=0..7)

Address offset: 0x118 + (n × 0x8)

Last results is equal or above CH[n].LIMIT.HIGH

Bit number	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
ID	A (bit 0)
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LIMITH			Last results is equal or above CH[n].LIMIT.HIGH
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.12.10.16 EVENTS_CH[n].LIMITL (n=0..7)

Address offset: 0x11C + (n × 0x8)

Last results is equal or below CH[n].LIMIT.LOW

Bit number	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
ID	A (bit 0)
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LIMITL			Last results is equal or below CH[n].LIMIT.LOW
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.12.10.17 PUBLISH_STARTED

Address offset: 0x180

Publish configuration for event STARTED

Bit number	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
ID	B (bit 31), A A A A (bits 3–0)
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event STARTED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.12.10.18 PUBLISH_END

Address offset: 0x184

Publish configuration for event END

Bit number	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
ID	B (bit 31), A A A A (bits 3–0)
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event END will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.12.10.19 PUBLISH_DONE

Address offset: 0x188

Publish configuration for event DONE

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event DONE will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.12.10.20 PUBLISH_RESULTDONE

Address offset: 0x18C

Publish configuration for event RESULTDONE

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event RESULTDONE will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.12.10.21 PUBLISH_CALIBRATEDONE

Address offset: 0x190

Publish configuration for event CALIBRATEDONE

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event CALIBRATEDONE will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.12.10.22 PUBLISH_STOPPED

Address offset: 0x194

Publish configuration for event STOPPED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event STOPPED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.12.10.23 PUBLISH_CH[n].LIMITH (n=0..7)

Address offset: 0x198 + (n × 0x8)

Publish configuration for event CH[n].LIMITH

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event CH[n].LIMITH will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.12.10.24 PUBLISH_CH[n].LIMITL (n=0..7)

Address offset: 0x19C + (n × 0x8)

Publish configuration for event CH[n].LIMITL

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event CH[n].LIMITL will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.12.10.25 INTEN

Address offset: 0x300

Enable or disable interrupt

Bit number	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
ID											V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	STARTED			Enable or disable interrupt for event STARTED
			Disabled	0	Disable
			Enabled	1	Enable
B	RW	END			Enable or disable interrupt for event END
			Disabled	0	Disable
			Enabled	1	Enable

Bit number	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
ID											V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
C	RW	DONE			Enable or disable interrupt for event DONE
			Disabled	0	Disable
			Enabled	1	Enable
D	RW	RESULTDONE			Enable or disable interrupt for event RESULTDONE
			Disabled	0	Disable
			Enabled	1	Enable
E	RW	CALIBRATEDONE			Enable or disable interrupt for event CALIBRATEDONE
			Disabled	0	Disable
			Enabled	1	Enable
F	RW	STOPPED			Enable or disable interrupt for event STOPPED
			Disabled	0	Disable
			Enabled	1	Enable
G	RW	CH0LIMITH			Enable or disable interrupt for event CH0LIMITH
			Disabled	0	Disable
			Enabled	1	Enable
H	RW	CH0LIMITL			Enable or disable interrupt for event CH0LIMITL
			Disabled	0	Disable
			Enabled	1	Enable
I	RW	CH1LIMITH			Enable or disable interrupt for event CH1LIMITH
			Disabled	0	Disable
			Enabled	1	Enable
J	RW	CH1LIMITL			Enable or disable interrupt for event CH1LIMITL
			Disabled	0	Disable
			Enabled	1	Enable
K	RW	CH2LIMITH			Enable or disable interrupt for event CH2LIMITH
			Disabled	0	Disable
			Enabled	1	Enable
L	RW	CH2LIMITL			Enable or disable interrupt for event CH2LIMITL
			Disabled	0	Disable
			Enabled	1	Enable
M	RW	CH3LIMITH			Enable or disable interrupt for event CH3LIMITH
			Disabled	0	Disable
			Enabled	1	Enable
N	RW	CH3LIMITL			Enable or disable interrupt for event CH3LIMITL
			Disabled	0	Disable
			Enabled	1	Enable
O	RW	CH4LIMITH			Enable or disable interrupt for event CH4LIMITH
			Disabled	0	Disable
			Enabled	1	Enable
P	RW	CH4LIMITL			Enable or disable interrupt for event CH4LIMITL
			Disabled	0	Disable
			Enabled	1	Enable
Q	RW	CH5LIMITH			Enable or disable interrupt for event CH5LIMITH
			Disabled	0	Disable
			Enabled	1	Enable
R	RW	CH5LIMITL			Enable or disable interrupt for event CH5LIMITL
			Disabled	0	Disable
			Enabled	1	Enable
S	RW	CH6LIMITH			Enable or disable interrupt for event CH6LIMITH
			Disabled	0	Disable

Bit number	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
ID											V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			Enabled	1	Enable
T	RW	CH6LIMITL			Enable or disable interrupt for event CH6LIMITL
			Disabled	0	Disable
			Enabled	1	Enable
U	RW	CH7LIMITH			Enable or disable interrupt for event CH7LIMITH
			Disabled	0	Disable
			Enabled	1	Enable
V	RW	CH7LIMITL			Enable or disable interrupt for event CH7LIMITL
			Disabled	0	Disable
			Enabled	1	Enable

6.12.10.26 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID											V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	STARTED			Write '1' to enable interrupt for event STARTED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	END			Write '1' to enable interrupt for event END
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	DONE			Write '1' to enable interrupt for event DONE
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
D	RW	RESULTDONE			Write '1' to enable interrupt for event RESULTDONE
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
E	RW	CALIBRATEDONE			Write '1' to enable interrupt for event CALIBRATEDONE
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
F	RW	STOPPED			Write '1' to enable interrupt for event STOPPED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
G	RW	CH0LIMITH			Write '1' to enable interrupt for event CH0LIMITH
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
H	RW	CH0LIMITL			Write '1' to enable interrupt for event CH0LIMITL

Bit number				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
ID				V U T S R Q P O N M L K J I H G F E D C B A
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
ID	Acce	Field	Value ID	Value	Description
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
I	RW	CH1LIMITH			Write '1' to enable interrupt for event CH1LIMITH
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
J	RW	CH1LIMITL			Write '1' to enable interrupt for event CH1LIMITL
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
K	RW	CH2LIMITH			Write '1' to enable interrupt for event CH2LIMITH
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
L	RW	CH2LIMITL			Write '1' to enable interrupt for event CH2LIMITL
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
M	RW	CH3LIMITH			Write '1' to enable interrupt for event CH3LIMITH
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
N	RW	CH3LIMITL			Write '1' to enable interrupt for event CH3LIMITL
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
O	RW	CH4LIMITH			Write '1' to enable interrupt for event CH4LIMITH
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
P	RW	CH4LIMITL			Write '1' to enable interrupt for event CH4LIMITL
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
Q	RW	CH5LIMITH			Write '1' to enable interrupt for event CH5LIMITH
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
R	RW	CH5LIMITL			Write '1' to enable interrupt for event CH5LIMITL
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
S	RW	CH6LIMITH			Write '1' to enable interrupt for event CH6LIMITH
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
T	RW	CH6LIMITL			Write '1' to enable interrupt for event CH6LIMITL
			Set	1	Enable
			Disabled	0	Read: Disabled

Bit number	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
ID											V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			Enabled	1	Read: Enabled
U	RW	CH7LIMITH			Write '1' to enable interrupt for event CH7LIMITH
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
V	RW	CH7LIMITL			Write '1' to enable interrupt for event CH7LIMITL
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.12.10.27 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID											V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	STARTED			Write '1' to disable interrupt for event STARTED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	END			Write '1' to disable interrupt for event END
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	DONE			Write '1' to disable interrupt for event DONE
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
D	RW	RESULTDONE			Write '1' to disable interrupt for event RESULTDONE
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
E	RW	CALIBRATEDONE			Write '1' to disable interrupt for event CALIBRATEDONE
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
F	RW	STOPPED			Write '1' to disable interrupt for event STOPPED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
G	RW	CH0LIMITH			Write '1' to disable interrupt for event CH0LIMITH
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
H	RW	CH0LIMITL			Write '1' to disable interrupt for event CH0LIMITL
			Clear	1	Disable

Bit number				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
ID				V U T S R Q P O N M L K J I H G F E D C B A
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
ID	Acce	Field	Value ID	Value	Description
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
I	RW	CH1LIMITH			Write '1' to disable interrupt for event CH1LIMITH
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
J	RW	CH1LIMITL			Write '1' to disable interrupt for event CH1LIMITL
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
K	RW	CH2LIMITH			Write '1' to disable interrupt for event CH2LIMITH
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
L	RW	CH2LIMITL			Write '1' to disable interrupt for event CH2LIMITL
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
M	RW	CH3LIMITH			Write '1' to disable interrupt for event CH3LIMITH
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
N	RW	CH3LIMITL			Write '1' to disable interrupt for event CH3LIMITL
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
O	RW	CH4LIMITH			Write '1' to disable interrupt for event CH4LIMITH
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
P	RW	CH4LIMITL			Write '1' to disable interrupt for event CH4LIMITL
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
Q	RW	CH5LIMITH			Write '1' to disable interrupt for event CH5LIMITH
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
R	RW	CH5LIMITL			Write '1' to disable interrupt for event CH5LIMITL
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
S	RW	CH6LIMITH			Write '1' to disable interrupt for event CH6LIMITH
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
T	RW	CH6LIMITL			Write '1' to disable interrupt for event CH6LIMITL
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

Bit number	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
ID											V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
U	RW	CH7LIMITH			Write '1' to disable interrupt for event CH7LIMITH
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
V	RW	CH7LIMITL			Write '1' to disable interrupt for event CH7LIMITL
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.12.10.28 STATUS

Address offset: 0x400

Status

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	STATUS			Status
			Ready	0	ADC is ready. No on-going conversion.
			Busy	1	ADC is busy. Single conversion in progress.

6.12.10.29 ENABLE

Address offset: 0x500

Enable or disable ADC

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ENABLE			Enable or disable ADC
			Disabled	0	Disable ADC
			Enabled	1	Enable ADC
					When enabled, the ADC will acquire access to the analog input pins specified in the CH[n].PSELP and CH[n].PSELN registers.

6.12.10.30 CH[n].PSELP (n=0..7)

Address offset: 0x510 + (n × 0x10)

Input positive pin selection for CH[n]

Bit number	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
ID																												A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PSELP			Analog positive input channel
			NC	0	Not connected
			AnalogInput0	1	AIN0
			AnalogInput1	2	AIN1
			AnalogInput2	3	AIN2
			AnalogInput3	4	AIN3
			AnalogInput4	5	AIN4
			AnalogInput5	6	AIN5
			AnalogInput6	7	AIN6
			AnalogInput7	8	AIN7
			VDDGPIO	9	VDD_GPIO

6.12.10.31 CH[n].PSELN (n=0..7)

Address offset: 0x514 + (n × 0x10)

Input negative pin selection for CH[n]

Bit number	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
ID																												A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PSELN			Analog negative input, enables differential channel
			NC	0	Not connected
			AnalogInput0	1	AIN0
			AnalogInput1	2	AIN1
			AnalogInput2	3	AIN2
			AnalogInput3	4	AIN3
			AnalogInput4	5	AIN4
			AnalogInput5	6	AIN5
			AnalogInput6	7	AIN6
			AnalogInput7	8	AIN7
			VDD_GPIO	9	VDD_GPIO

6.12.10.32 CH[n].CONFIG (n=0..7)

Address offset: 0x518 + (n × 0x10)

Input configuration for CH[n]

Bit number	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
ID								G				F		E	E	E				D		C	C	C			B	B			A	A
Reset 0x00020000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	RESP			Positive channel resistor control
			Bypass	0	Bypass resistor ladder
			Pulldown	1	Pull-down to GND
			Pullup	2	Pull-up to VDD_GPIO
			VDD1_2	3	Set input at VDD_GPIO/2
B	RW	RESN			Negative channel resistor control
			Bypass	0	Bypass resistor ladder

Bit number	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
ID								G				F		E	E	E				D		C	C	C			B	B			A	A
Reset 0x00020000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			Pulldown	1	Pull-down to GND
			Pullup	2	Pull-up to VDD_GPIO
			VDD1_2	3	Set input at VDD_GPIO/2
C	RW	GAIN			Gain control
			Gain1_6	0	1/6
			Gain1_5	1	1/5
			Gain1_4	2	1/4
			Gain1_3	3	1/3
			Gain1_2	4	1/2
			Gain1	5	1
			Gain2	6	2
			Gain4	7	4
D	RW	REFSEL			Reference control
			Internal	0	Internal reference (0.6 V)
			VDD1_4	1	VDD_GPIO/4 as reference
E	RW	TACQ			Acquisition time, the time the ADC uses to sample the input voltage
			3us	0	3 us
			5us	1	5 us
			10us	2	10 us
			15us	3	15 us
			20us	4	20 us
			40us	5	40 us
F	RW	MODE			Enable differential mode
			SE	0	Single ended, PSELN will be ignored, negative input to ADC shorted to GND
			Diff	1	Differential
G	RW	BURST			Enable burst mode
			Disabled	0	Burst mode is disabled (normal operation)
			Enabled	1	Burst mode is enabled. SAADC takes 2^OVERSAMPLE number of samples as fast as it can, and sends the average to Data RAM.

6.12.10.33 CH[n].LIMIT (n=0..7)

Address offset: 0x51C + (n × 0x10)

High/low limits for event monitoring a channel

Bit number	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
ID	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x7FFF8000	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LOW		[-32768 to +32767]	Low level limit
B	RW	HIGH		[-32768 to +32767]	High level limit

6.12.10.34 RESOLUTION

Address offset: 0x5F0

Resolution configuration

Bit number	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
ID																														A	A	A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	VAL			Set the resolution
			8bit	0	8 bit
			10bit	1	10 bit
			12bit	2	12 bit
			14bit	3	14 bit

6.12.10.35 OVERSAMPLE

Address offset: 0x5F4

Oversampling configuration. OVERSAMPLE should not be combined with SCAN. The RESOLUTION is applied before averaging, thus for high OVERSAMPLE a higher RESOLUTION should be used.

Bit number	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
ID																													A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	OVERSAMPLE			Oversample control
			Bypass	0	Bypass oversampling
			Over2x	1	Oversample 2x
			Over4x	2	Oversample 4x
			Over8x	3	Oversample 8x
			Over16x	4	Oversample 16x
			Over32x	5	Oversample 32x
			Over64x	6	Oversample 64x
			Over128x	7	Oversample 128x
			Over256x	8	Oversample 256x

6.12.10.36 SAMPLERATE

Address offset: 0x5F8

Controls normal or continuous sample rate

Bit number	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
ID																				B		A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CC		[80..2047]	Capture and compare value. Sample rate is 16 MHz/CC
B	RW	MODE			Select mode for sample rate control
			Task	0	Rate is controlled from SAMPLE task
			Timers	1	Rate is controlled from local timer (use CC to control the rate)

6.12.10.37 RESULT.PTR

Address offset: 0x62C

Data pointer

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			Data pointer

Note: See the memory chapter for details about which memories are available for EasyDMA.

6.12.10.38 RESULT.MAXCNT

Address offset: 0x630

Maximum number of buffer words to transfer

Bit number	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
ID																		A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXCNT			Maximum number of buffer words to transfer

6.12.10.39 RESULT.AMOUNT

Address offset: 0x634

Number of buffer words transferred since last START

Bit number	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
ID																		A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	AMOUNT			Number of buffer words transferred since last START. This register can be read after an END or STOPPED event.

6.12.11 Electrical specification

6.12.11.1 SAADC Electrical Specification

Symbol	Description	Min.	Typ.	Max.	Units
DNL_{10}	Differential non-linearity, 10-bit resolution	-0.95	<1		LSB10b
INL_{10}	Integral non-linearity, 10-bit resolution		1		LSB1(
V_{OS}	Differential offset error (calibrated), 10-bit resolution [a]		±2		LSB10b
C_{EG}	Gain error temperature coefficient	-0.05	0.02	0.05	%/◦C
f_{SAMPLE}	Maximum sampling rate			200	kHz
$t_{ACQ,10k}$	Acquisition time (configurable), source Resistance <= 10kOhm		3		µs
$t_{ACQ,40k}$	Acquisition time (configurable), source Resistance <= 40kOhm		5		µs
$t_{ACQ,100k}$	Acquisition time (configurable), source Resistance <= 100kOhm		10		µs

[a] Digital output code at zero volt differential input.

Symbol	Description	Min.	Typ.	Max.	Units
$t_{ACQ,200k}$	Acquisition time (configurable), source Resistance <= 200kOhm		15		µs
$t_{ACQ,400k}$	Acquisition time (configurable), source Resistance <= 400kOhm		20		µs
$t_{ACQ,800k}$	Acquisition time (configurable), source Resistance <= 800kOhm		40		µs
t_{CONV}	Conversion time		<2		µs
$E_{G1/6}$	Error[b] for Gain = 1/6	-3		3	%
$E_{G1/4}$	Error[b] for Gain = 1/4	-3		3	%
$E_{G1/2}$	Error[b] for Gain = 1/2	-3		4	%
E_{G1}	Error[b] for Gain = 1	-3		4	%
C_{SAMPLE}	Sample and hold capacitance at maximum gain[9]		2.5		pF
R_{INPUT}	Input resistance		>1		MΩ
E_{NOB}	Effective number of bits, differential mode, 12-bit resolution, 1/1 gain, 3 µs acquisition time, HFXO, 200 ksps		9		Bit
S_{NDR}	Peak signal to noise and distortion ratio, differential mode, 12-bit resolution, 1/1 gain, 3 µs acquisition time, HFXO, 200 ksps		56		dB
S_{FDR}	Spurious free dynamic range, differential mode, 12-bit resolution, 1/1 gain, 3 µs acquisition time, HFXO, 200 ksps		70		dBc
R_{LADDER}	Ladder resistance		160		kΩ

6.12.12 Performance factors

Clock jitter, affecting sample timing accuracy, and circuit noise can affect ADC performance.

Jitter can be between START tasks or from START task to acquisition. START timer accuracy and startup times of regulators and references will contribute to variability. Sources of circuit noise may include CPU activity and the DC/DC regulator. Best ADC performance is achieved using START timing based on the TIMER module, HFXO clock source, and Constant Latency mode.

6.13 SPIM — Serial peripheral interface master with EasyDMA

The SPI master can communicate with multiple slaves using individual chip select signals for each of the slave devices attached to a bus.

Listed here are the main features for the SPIM

- SPI mode 0-3
- EasyDMA direct transfer to/from RAM for both SPI Slave and SPI Master
- Individual selection of IO pin for each SPI signal

b Does not include temperature drift
9 Maximum gain corresponds to highest capacitance.

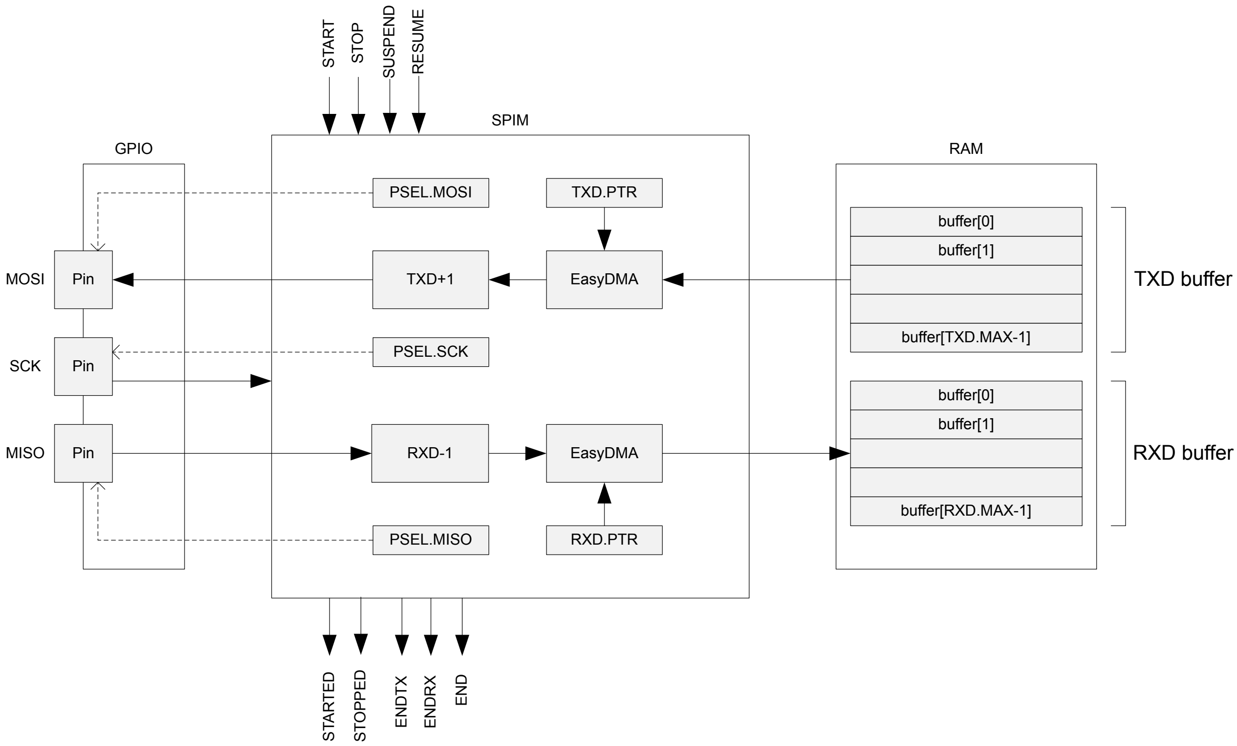

Figure 71: SPIM — SPI master with EasyDMA

The SPIM does not implement support for chip select directly. Therefore, the CPU must use available GPIOs to select the correct slave and control this independently of the SPI master. The SPIM supports SPI modes 0 through 3. The CONFIG register allows setting CPOL and CPHA appropriately.

Mode	Clock polarity	Clock phase
	CPOL	CPHA
SPI_MODE0	0 (Active High)	0 (Leading)
SPI_MODE1	0 (Active High)	1 (Trailing)
SPI_MODE2	1 (Active Low)	0 (Leading)
SPI_MODE3	1 (Active Low)	1 (Trailing)

Table 69: SPI modes

6.13.1 SPI master transaction sequence

An SPI master transaction consists of a sequence started by the START task followed by a number of events, and finally the STOP task.

An SPI master transaction is started by triggering the START task. The ENDTX event will be generated when the transmitter has transmitted all bytes in the TXD buffer as specified in the TXD.MAXCNT register. The ENDRX event will be generated when the receiver has filled the RXD buffer, i.e. received the last possible byte as specified in the RXD.MAXCNT register.

Following a START task, the SPI master will generate an END event when both ENDRX and ENDTX have been generated.

The SPI master is stopped by triggering the STOP task. A STOPPED event is generated when the SPI master has stopped.

If the ENDRX event has not already been generated when the SPI master has come to a stop, the SPI master will generate the ENDRX event explicitly even though the RX buffer is not full.

If the ENDTX event has not already been generated when the SPI master has come to a stop, the SPI master will generate the ENDTX event explicitly even though all bytes in the TXD buffer, as specified in the TXD.MAXCNT register, have not been transmitted.

The SPI master is a synchronous interface, and for every byte that is sent, a different byte will be received at the same time; this is illustrated in SPI master transaction on page 228.

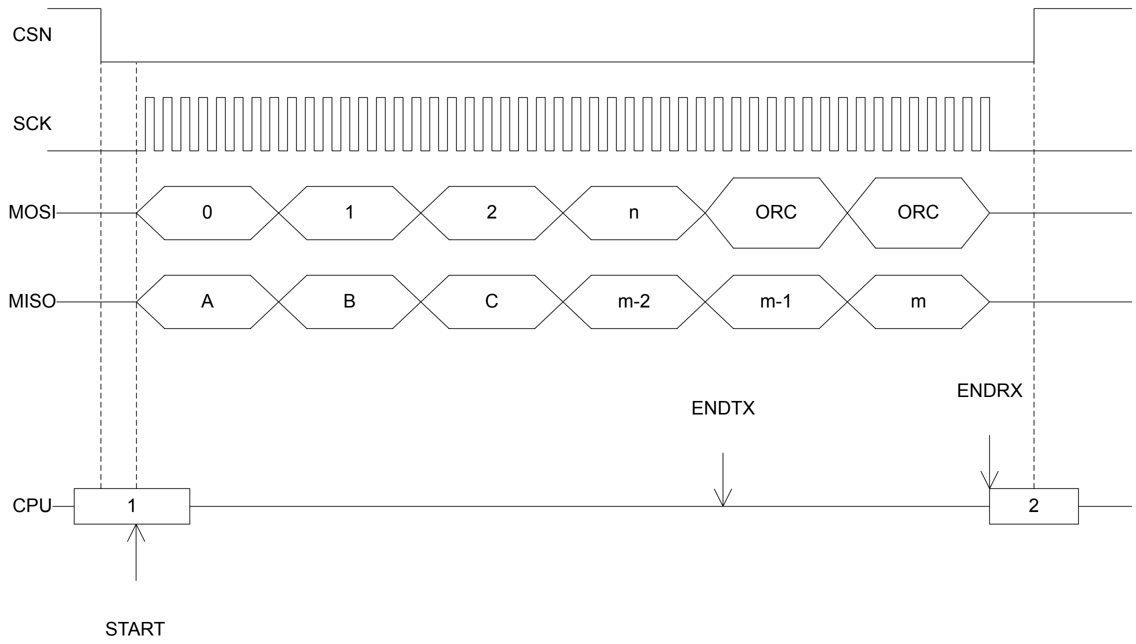

Figure 72: SPI master transaction

6.13.2 Master mode pin configuration

The SCK, MOSI, and MISO signals associated with the SPI master are mapped to physical pins according to the configuration specified in the PSEL.SCK, PSEL.MOSI, and PSEL.MISO registers respectively.

The PSEL.SCK, PSEL.MOSI, and PSEL.MISO registers and their configurations are only used as long as the SPI master is enabled, and retained only as long as the device is in ON mode. PSEL.SCK, PSEL.MOSI and PSEL.MISO must only be configured when the SPI master is disabled.

To secure correct behavior in the SPI, the pins used by the SPI must be configured in the GPIO peripheral as described in GPIO configuration on page 228 prior to enabling the SPI. This configuration must be retained in the GPIO for the selected IOs as long as the SPI is enabled.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

SPI master signal	SPI master pin	Direction	Output value
SCK	As specified in PSEL.SCK	Output	Same as CONFIG.CPOL
MOSI	As specified in PSEL.MOSI	Output	0
MISO	As specified in PSEL.MISO	Input	Not applicable

Table 70: GPIO configuration

6.13.3 Shared resources

The SPI shares registers and other resources with other peripherals that have the same ID as the SPI. Therefore, the user must disable all peripherals that have the same ID as the SPI before the SPI can be configured and used.

Disabling a peripheral that has the same ID as the SPI will not reset any of the registers that are shared with the SPI. It is therefore important to configure all relevant SPI registers explicitly to secure that it operates correctly.

See the Instantiation table in Instantiation on page 23 for details on peripherals and their IDs.

6.13.4 EasyDMA

The SPIM implements EasyDMA for accessing RAM without CPU involvement.

The SPIM peripheral implements the following EasyDMA channels:

Channel	Type	Register Cluster
TXD	READER	TXD
RXD	WRITER	RXD

Table 71: SPIM EasyDMA Channels

For detailed information regarding the use of EasyDMA, see EasyDMA on page 44.

The .PTR and .MAXCNT registers are double-buffered. They can be updated and prepared for the next transmission immediately after having received the STARTED event.

The SPI master will automatically stop transmitting after TXD.MAXCNT bytes have been transmitted and RXD.MAXCNT bytes have been received. If RXD.MAXCNT is larger than TXD.MAXCNT, the remaining transmitted bytes will contain the value defined in the ORC register. If TXD.MAXCNT is larger than RXD.MAXCNT, the superfluous received bytes will be discarded.

The ENDRX/ENDTX event indicate that EasyDMA has finished accessing respectively the RX/TX buffer in RAM. The END event gets generated when both RX and TX are finished accessing the buffers in RAM.

In the case of bus congestion as described in AHB multilayer interconnect on page 47, data loss may occur.

6.13.5 Low power

When putting the system in low power and the peripheral is not needed, lowest possible power consumption is achieved by stopping, and then disabling the peripheral.

The STOP task may not be always needed (the peripheral might already be stopped), but if it is sent, software shall wait until the STOPPED event was received as a response before disabling the peripheral through the ENABLE register.

6.13.6 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50008000 0x40008000	SPIM	SPIM0 : S SPIM0 : NS	US	SA	SPI master 0	
0x50009000 0x40009000	SPIM	SPIM1 : S SPIM1 : NS	US	SA	SPI master 1	
0x5000A000 0x4000A000	SPIM	SPIM2 : S SPIM2 : NS	US	SA	SPI master 2	
0x5000B000 0x4000B000	SPIM	SPIM3 : S SPIM3 : NS	US	SA	SPI master 3	

Table 72: Instances

Register	Offset	Security	Description
TASKS_START	0x010		Start SPI transaction
TASKS_STOP	0x014		Stop SPI transaction
TASKS_SUSPEND	0x01C		Suspend SPI transaction
TASKS_RESUME	0x020		Resume SPI transaction
SUBSCRIBE_START	0x090		Subscribe configuration for task START
SUBSCRIBE_STOP	0x094		Subscribe configuration for task STOP
SUBSCRIBE_SUSPEND	0x09C		Subscribe configuration for task SUSPEND
SUBSCRIBE_RESUME	0x0A0		Subscribe configuration for task RESUME
EVENTS_STOPPED	0x104		SPI transaction has stopped
EVENTS_ENDRX	0x110		End of RXD buffer reached
EVENTS_END	0x118		End of RXD buffer and TXD buffer reached
EVENTS_ENDTX	0x120		End of TXD buffer reached
EVENTS_STARTED	0x14C		Transaction started
PUBLISH_STOPPED	0x184		Publish configuration for event STOPPED
PUBLISH_ENDRX	0x190		Publish configuration for event ENDRX
PUBLISH_END	0x198		Publish configuration for event END
PUBLISH_ENDTX	0x1A0		Publish configuration for event ENDTX
PUBLISH_STARTED	0x1CC		Publish configuration for event STARTED
SHORTS	0x200		Shortcuts between local events and tasks
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
ENABLE	0x500		Enable SPIM
PSEL.SCK	0x508		Pin select for SCK
PSEL.MOSI	0x50C		Pin select for MOSI signal
PSEL.MISO	0x510		Pin select for MISO signal
FREQUENCY	0x524		SPI frequency. Accuracy depends on the HFCLK source selected.
RXD.PTR	0x534		Data pointer
RXD.MAXCNT	0x538		Maximum number of bytes in receive buffer
RXD.AMOUNT	0x53C		Number of bytes transferred in the last transaction
RXD.LIST	0x540		EasyDMA list type
TXD.PTR	0x544		Data pointer
TXD.MAXCNT	0x548		Maximum number of bytes in transmit buffer
TXD.AMOUNT	0x54C		Number of bytes transferred in the last transaction
TXD.LIST	0x550		EasyDMA list type
CONFIG	0x554		Configuration register
ORC	0x5C0		Over-read character. Character clocked out in case an over-read of the TXD buffer.

Table 73: Register overview

6.13.6.1 TASKS_START

Address offset: 0x010

Start SPI transaction

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_START			Start SPI transaction
			Trigger	1	Trigger task

6.13.6.2 TASKS_STOP

Address offset: 0x014

Stop SPI transaction

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_STOP			Stop SPI transaction
			Trigger	1	Trigger task

6.13.6.3 TASKS_SUSPEND

Address offset: 0x01C

Suspend SPI transaction

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_SUSPEND			Suspend SPI transaction
			Trigger	1	Trigger task

6.13.6.4 TASKS_RESUME

Address offset: 0x020

Resume SPI transaction

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_RESUME			Resume SPI transaction
			Trigger	1	Trigger task

6.13.6.5 SUBSCRIBE_START

Address offset: 0x090

Subscribe configuration for task START

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task START will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.13.6.6 SUBSCRIBE_STOP

Address offset: 0x094

Subscribe configuration for task STOP

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STOP will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.13.6.7 SUBSCRIBE_SUSPEND

Address offset: 0x09C

Subscribe configuration for task SUSPEND

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task SUSPEND will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.13.6.8 SUBSCRIBE_RESUME

Address offset: 0x0A0

Subscribe configuration for task RESUME

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task RESUME will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.13.6.9 EVENTS_STOPPED

Address offset: 0x104

SPI transaction has stopped

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_STOPPED			SPI transaction has stopped
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.13.6.10 EVENTS_ENDRX

Address offset: 0x110

End of RXD buffer reached

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_ENDRX			End of RXD buffer reached
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.13.6.11 EVENTS_END

Address offset: 0x118

End of RXD buffer and TXD buffer reached

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_END			End of RXD buffer and TXD buffer reached
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.13.6.12 EVENTS_ENDTX

Address offset: 0x120

End of TXD buffer reached

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_ENDTX			End of TXD buffer reached
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.13.6.13 EVENTS_STARTED

Address offset: 0x14C

Transaction started

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_STARTED			Transaction started
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.13.6.14 PUBLISH_STOPPED

Address offset: 0x184

Publish configuration for event STOPPED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event STOPPED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.13.6.15 PUBLISH_ENDRX

Address offset: 0x190

Publish configuration for event ENDRX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event ENDRX will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.13.6.16 PUBLISH_END

Address offset: 0x198

Publish configuration for event END

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event END will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.13.6.17 PUBLISH_ENDTX

Address offset: 0x1A0

Publish configuration for event ENDTX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event ENDTX will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.13.6.18 PUBLISH_STARTED

Address offset: 0x1CC

Publish configuration for event STARTED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event STARTED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.13.6.19 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

Bit number	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
ID															A																	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	END_START			Shortcut between event END and task START
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut

6.13.6.20 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID													E											D		C		B			A	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	STOPPED			Write '1' to enable interrupt for event STOPPED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	ENDRX			Write '1' to enable interrupt for event ENDRX
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	END			Write '1' to enable interrupt for event END
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
D	RW	ENDTX			Write '1' to enable interrupt for event ENDTX
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
E	RW	STARTED			Write '1' to enable interrupt for event STARTED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.13.6.21 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID													E											D		C		B			A	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	STOPPED			Write '1' to disable interrupt for event STOPPED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	ENDRX			Write '1' to disable interrupt for event ENDRX
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	END			Write '1' to disable interrupt for event END
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
D	RW	ENDTX			Write '1' to disable interrupt for event ENDTX
			Clear	1	Disable

Bit number	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
ID													E											D		C		B			A	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
E	RW	STARTED			Write '1' to disable interrupt for event STARTED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.13.6.22 ENABLE

Address offset: 0x500

Enable SPIM

Bit number	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
ID																													A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ENABLE			Enable or disable SPIM
			Disabled	0	Disable SPIM
			Enabled	7	Enable SPIM

6.13.6.23 PSEL.SCK

Address offset: 0x508

Pin select for SCK

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.13.6.24 PSEL.MOSI

Address offset: 0x50C

Pin select for MOSI signal

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.13.6.25 PSEL.MISO

Address offset: 0x510

Pin select for MISO signal

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.13.6.26 FREQUENCY

Address offset: 0x524

SPI frequency. Accuracy depends on the HFCLK source selected.

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x04000000	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	FREQUENCY			SPI master data rate
			K125	0x02000000	125 kbps
			K250	0x04000000	250 kbps
			K500	0x08000000	500 kbps
			M1	0x10000000	1 Mbps
			M2	0x20000000	2 Mbps
			M4	0x40000000	4 Mbps
			M8	0x80000000	8 Mbps

6.13.6.27 RXD.PTR

Address offset: 0x534

Data pointer

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			Data pointer

Note: See the memory chapter for details about which memories are available for EasyDMA.

6.13.6.28 RXD.MAXCNT

Address offset: 0x538

Maximum number of bytes in receive buffer

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXCNT		[1..0x1FFF]	Maximum number of bytes in receive buffer

6.13.6.29 RXD.AMOUNT

Address offset: 0x53C

Number of bytes transferred in the last transaction

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	AMOUNT		[1..0x1FFF]	Number of bytes transferred in the last transaction

6.13.6.30 RXD.LIST

Address offset: 0x540

EasyDMA list type

Bit number	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
ID																															A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LIST			List type
			Disabled	0	Disable EasyDMA list
			ArrayList	1	Use array list

6.13.6.31 TXD.PTR

Address offset: 0x544

Data pointer

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			Data pointer **Note:** See the memory chapter for details about which memories are available for EasyDMA.

6.13.6.32 TXD.MAXCNT

Address offset: 0x548

Maximum number of bytes in transmit buffer

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXCNT		[1..0x1FFF]	Maximum number of bytes in transmit buffer

6.13.6.33 TXD.AMOUNT

Address offset: 0x54C

Number of bytes transferred in the last transaction

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	R	AMOUNT		[1..0x1FFF]	Number of bytes transferred in the last transaction

6.13.6.34 TXD.LIST

Address offset: 0x550

EasyDMA list type

Bit number	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
ID																															A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	LIST			List type
			Disabled	0	Disable EasyDMA list
			ArrayList	1	Use array list

6.13.6.35 CONFIG

Address offset: 0x554

Configuration register

Bit number	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
ID																														C	B	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	ORDER			Bit order
			MsbFirst	0	Most significant bit shifted out first
			LsbFirst	1	Least significant bit shifted out first
B	RW	CPHA			Serial clock (SCK) phase
			Leading	0	Sample on leading edge of clock, shift serial data on trailing edge
			Trailing	1	Sample on trailing edge of clock, shift serial data on leading edge
C	RW	CPOL			Serial clock (SCK) polarity
			ActiveHigh	0	Active high
			ActiveLow	1	Active low

6.13.6.36 ORC

Address offset: 0x5C0

Over-read character. Character clocked out in case an over-read of the TXD buffer.

Bit number	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
ID	A A A A A A A A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ORC			Over-read character. Character clocked out in case an over-read of the TXD buffer.

6.13.7 Electrical specification

6.13.7.1 SPIM master interface electrical specifications

Symbol	Description	Min.	Typ.	Max.	Units
f_{SPIM}	Bit rates for SPIM[10]			8	Mbps
$t_{SPIM,START}$	Time from START task to transmission started		1		µs

6.13.7.2 Serial Peripheral Interface Master (SPIM) timing specifications

Symbol	Description	Min.	Typ.	Max.	Units
$t_{SPIM,CSCK}$	SCK period		125		ns
$t_{SPIM,RSCK,LD}$	SCK rise time, standard drive[a]			$t_{RF,25pF}$	
$t_{SPIM,RSCK,HD}$	SCK rise time, high drive[a]			$t_{HRF,25pF}$	
$t_{SPIM,FSCK,LD}$	SCK fall time, standard drive[a]			$t_{RF,25pF}$	
$t_{SPIM,FSCK,HD}$	SCK fall time, high drive[a]			$t_{HRF,25pF}$	
$t_{SPIM,WHSCK}$	SCK high time[a]	$(0.5*t_{CSCK}$ $– t_{RSCK}$			
$t_{SPIM,WLSCK}$	SCK low time[a]	$(0.5*t_{CSCK})$ $– t_{FSCK}$			
$t_{SPIM,SUMI}$	MISO to CLK edge setup time	19			ns
$t_{SPIM,HMI}$	CLK edge to MISO hold time	18			ns
$t_{SPIM,VMO}$	CLK edge to MOSI valid			59	ns
$t_{SPIM,HMO}$	MOSI hold time after CLK edge	20			ns

[10] High bit rates may require GPIOs to be set as High Drive, see GPIO chapter for more details.
[a] At 25pF load, including GPIO pin capacitance, see GPIO spec.

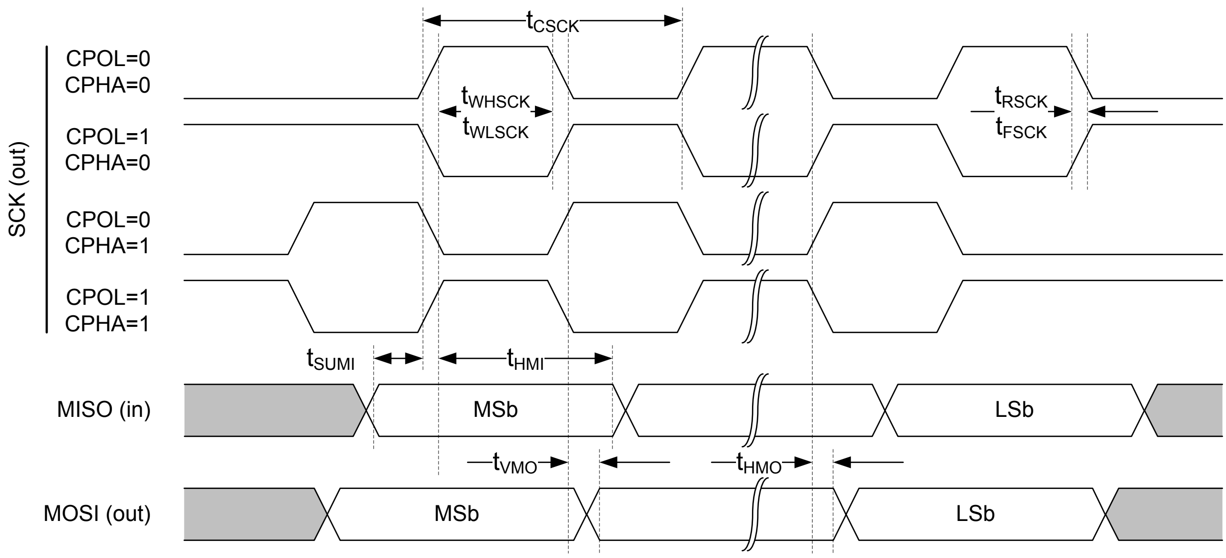

Figure 73: SPIM timing diagram

6.14 SPIS — Serial peripheral interface slave with EasyDMA

SPI slave (SPIS) is implemented with EasyDMA support for ultra low power serial communication from an external SPI master. EasyDMA in conjunction with hardware-based semaphore mechanisms removes all real-time requirements associated with controlling the SPI slave from a low priority CPU execution context.

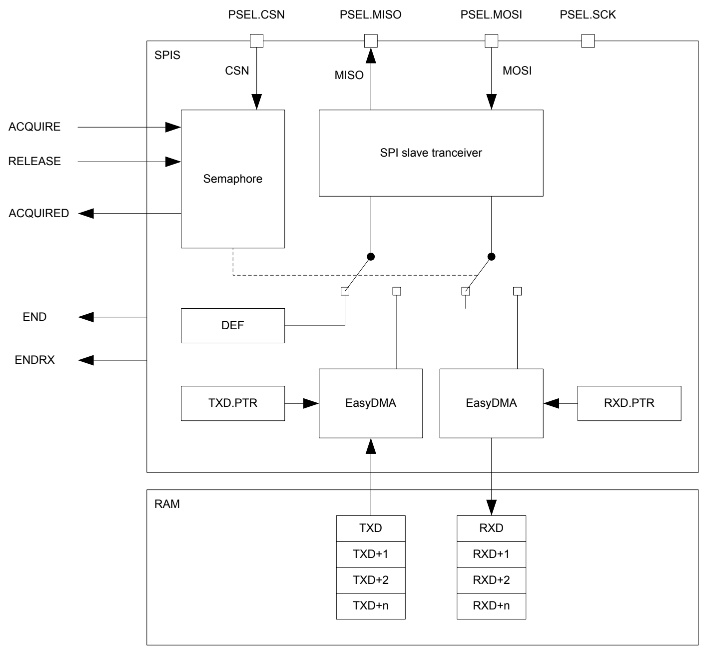

Figure 74: SPI slave

The SPIS supports SPI modes 0 through 3. The CONFIG register allows setting CPOL and CPHA appropriately.

Mode	Clock polarity	Clock phase
	CPOL	CPHA
SPI_MODE0	0 (Active High)	0 (Trailing Edge)
SPI_MODE1	0 (Active High)	1 (Leading Edge)
SPI_MODE2	1 (Active Low)	0 (Trailing Edge)
SPI_MODE3	1 (Active Low)	1 (Leading Edge)

Table 74: SPI modes

6.14.1 Shared resources

The SPI slave shares registers and other resources with other peripherals that have the same ID as the SPI slave. Therefore, you must disable all peripherals that have the same ID as the SPI slave before the SPI slave can be configured and used.

Disabling a peripheral that has the same ID as the SPI slave will not reset any of the registers that are shared with the SPI slave. It is important to configure all relevant SPI slave registers explicitly to secure that it operates correctly.

The Instantiation table in Instantiation on page 23 shows which peripherals have the same ID as the SPI slave.

6.14.2 EasyDMA

The SPIS implements EasyDMA for accessing RAM without CPU involvement.

The SPIS peripheral implements the following EasyDMA channels:

Channel	Type	Register Cluster
TXD	READER	TXD
RXD	WRITER	RXD

Table 75: SPIS EasyDMA Channels

For detailed information regarding the use of EasyDMA, see EasyDMA on page 44.

If RXD.MAXCNT is larger than TXD.MAXCNT, the remaining transmitted bytes will contain the value defined in the ORC register.

The END event indicates that EasyDMA has finished accessing the buffer in RAM.

6.14.3 SPI slave operation

SPI slave uses two memory pointers, RXD.PTR and TXD.PTR, that point to the RXD buffer (receive buffer) and TXD buffer (transmit buffer) respectively. Since these buffers are located in RAM, which can be accessed by both the SPI slave and the CPU, a hardware based semaphore mechanism is implemented to enable safe sharing.

Before the CPU can safely update the RXD.PTR and TXD.PTR pointers it must first acquire the SPI semaphore. The CPU can acquire the semaphore by triggering the ACQUIRE task and then receiving the ACQUIRED event. When the CPU has updated the RXD.PTR and TXD.PTR pointers the CPU must release the semaphore before the SPI slave will be able to acquire it.

The CPU releases the semaphore by triggering the RELEASE task, this is illustrated in SPI transaction when shortcut between END and ACQUIRE is enabled on page 244. Triggering the RELEASE task when the semaphore is not granted to the CPU will have no effect. See Semaphore operation on page 245 for more information

If the CPU is not able to reconfigure the TXD.PTR and RXD.PTR between granted transactions, the same TX data will be clocked out and the RX buffers will be overwritten. To prevent this from happening, the END_ACQUIRE shortcut can be used. With this shortcut enabled the semaphore will be handed over to the CPU automatically after the granted transaction has completed, giving the CPU the ability to update the TXPTR and RXPTR between every granted transaction.

The ENDRX event is generated when the RX buffer has been filled.

The RXD.MAXCNT register specifies the maximum number of bytes the SPI slave can receive in one granted transaction. If the SPI slave receives more than RXD.MAXCNT number of bytes, an OVERFLOW will be indicated in the STATUS register and the incoming bytes will be discarded.

The TXD.MAXCNT parameter specifies the maximum number of bytes the SPI slave can transmit in one granted transaction. If the SPI slave is forced to transmit more than TXD.MAXCNT number of bytes, an OVERREAD will be indicated in the STATUS register and the ORC character will be clocked out.

The RXD.AMOUNT and TXD.AMOUNT registers are updated when a granted transaction is completed. The TXD.AMOUNT register indicates how many bytes were read from the TX buffer in the last transaction, that is, ORC (over-read) characters are not included in this number. Similarly, the RXD.AMOUNT register indicates how many bytes were written into the RX buffer in the last transaction.

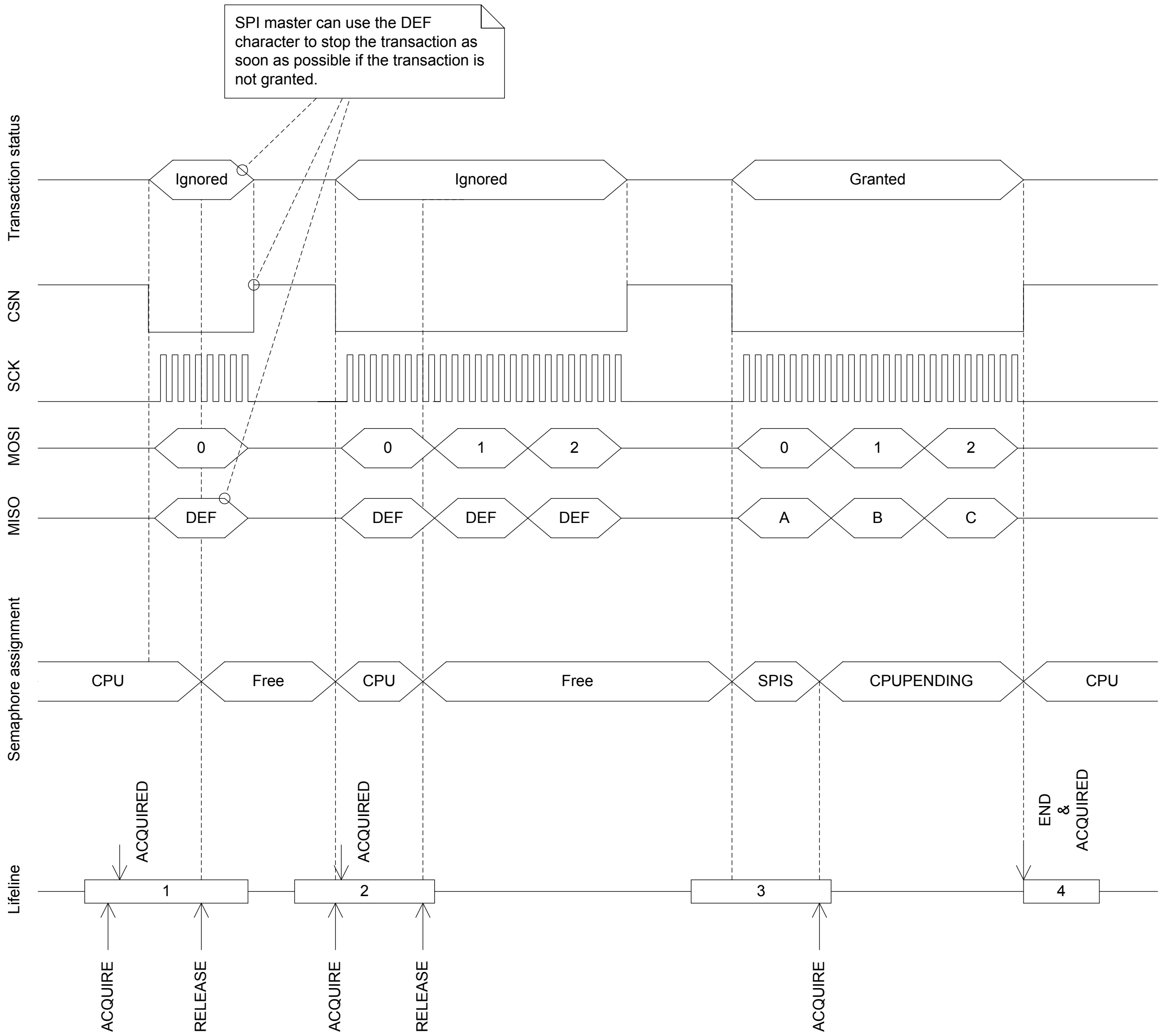

Figure 75: SPI transaction when shortcut between END and ACQUIRE is enabled

6.14.4 Semaphore operation

The semaphore is a mechanism implemented inside the SPI slave that prevents simultaneous access to the data buffers by the SPI slave and CPU.

The semaphore is by default assigned to the CPU after the SPI slave is enabled. No ACQUIRED event will be generated for this initial semaphore handover. An ACQUIRED event will be generated immediately if the ACQUIRE task is triggered while the semaphore is assigned to the CPU. The figure SPI semaphore FSM on page 245 illustrates the transitions between states in the semaphore based on the relevant tasks and events.

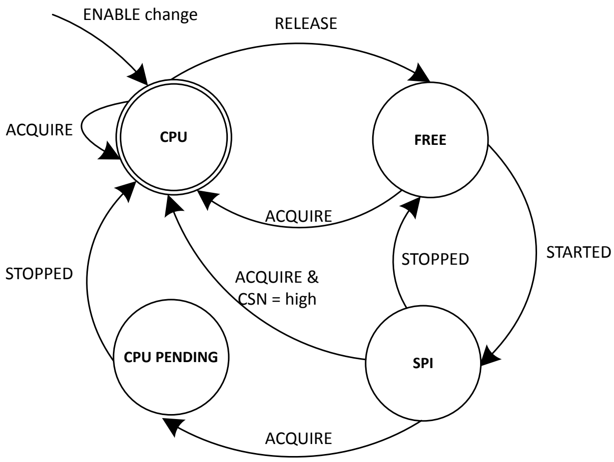

Figure 76: SPI semaphore FSM

> **Note:** The semaphore mechanism does not, at any time, prevent the CPU from performing read or write access to the RXD.PTR register, the TXD.PTR registers, or the RAM that these pointers are pointing to. The semaphore is only telling when these can be updated by the CPU so that safe sharing is achieved.

The SPI slave will try to acquire the semaphore when STARTED event is detected, the even also indicates that CSN is currently low. If the SPI slave does not manage to acquire the semaphore at this point (i.e., it is under CPU's control), the transaction will be ignored. This means that all incoming data on MOSI will be discarded, and the DEF (default) character will be clocked out on the MISO line throughout the whole transaction. This will also be the case even if the semaphore is released by the CPU during the transaction. In case of a race condition where the CPU and the SPI slave try to acquire the semaphore at the same time, as illustrated in lifeline item 2 in figure SPI transaction when shortcut between END and ACQUIRE is enabled on page 244, the semaphore will be granted to the CPU.

If the SPI slave acquires the semaphore, the transaction will be granted. The incoming data on MOSI will be stored in the RXD buffer and the data in the TXD buffer will be clocked out on MISO.

When a granted transaction is completed and CSN goes high, the SPI slave will automatically release the semaphore and generate the END event.

As long as the semaphore is available, the SPI slave can be granted multiple transactions one after the other.

If the CPU tries to acquire the semaphore while it is assigned to the SPI slave, an immediate handover will not be granted. However, the semaphore will be handed over to the CPU as soon as the SPI slave has released the semaphore after the granted transaction is completed. If the END_ACQUIRE shortcut is enabled and the CPU has triggered the ACQUIRE task during a granted transaction, only one ACQUIRE request will be served following the END event.

6.14.5 Pin configuration

The CSN, SCK, MOSI, and MISO signals associated with the SPI slave are mapped to physical pins according to the configuration specified in the PSEL.CSN, PSEL.SCK, PSEL.MOSI, and PSEL.MISO registers respectively. If the CONNECT field of any of these registers is set to Disconnected, the associated SPI slave signal will not be connected to any physical pins.

The PSEL.CSN, PSEL.SCK, PSEL.MOSI, and PSEL.MISO registers and their configurations are only used as long as the SPI slave is enabled, and retained only as long as the device is in System ON mode, see POWER — Power control on page 59 chapter for more information about power modes. When the peripheral is disabled, the pins will behave as regular GPIOs, and use the configuration in their respective OUT bit field and PIN_CNF[n] register. PSEL.CSN, PSEL.SCK, PSEL.MOSI, and PSEL.MISO must only be configured when the SPI slave is disabled.

To secure correct behavior in the SPI slave, the pins used by the SPI slave must be configured in the GPIO peripheral as described in GPIO configuration before enabling peripheral on page 246 before enabling the SPI slave. This is to secure that the pins used by the SPI slave are driven correctly if the SPI slave itself is temporarily disabled, or if the device temporarily enters System OFF. This configuration must be retained in the GPIO for the selected I/Os as long as the SPI slave is to be recognized by an external SPI master.

The MISO line is set in high impedance as long as the SPI slave is not selected with CSN.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

SPI signal	SPI pin	Direction	Output value	Comment
CSN	As specified in PSEL.CSN	Input	Not applicable	
SCK	As specified in PSEL.SCK	Input	Not applicable	
MOSI	As specified in PSEL.MOSI	Input	Not applicable	
MISO	As specified in PSEL.MISO	Input	Not applicable	Emulates that the SPI slave is not selected.

Table 76: GPIO configuration before enabling peripheral

6.14.6 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50008000 0x40008000	SPIS	SPIS0 : S SPIS0 : NS	US	SA	SPI slave 0	
0x50009000 0x40009000	SPIS	SPIS1 : S SPIS1 : NS	US	SA	SPI slave 1	
0x5000A000 0x4000A000	SPIS	SPIS2 : S SPIS2 : NS	US	SA	SPI slave 2	
0x5000B000 0x4000B000	SPIS	SPIS3 : S SPIS3 : NS	US	SA	SPI slave 3	

Table 77: Instances

Register	Offset	Security	Description
TASKS_ACQUIRE	0x024		Acquire SPI semaphore
TASKS_RELEASE	0x028		Release SPI semaphore, enabling the SPI slave to acquire it
SUBSCRIBE_ACQUIRE	0x0A4		Subscribe configuration for task ACQUIRE
SUBSCRIBE_RELEASE	0x0A8		Subscribe configuration for task RELEASE
EVENTS_END	0x104		Granted transaction completed
EVENTS_ENDRX	0x110		End of RXD buffer reached
EVENTS_ACQUIRED	0x128		Semaphore acquired

Register	Offset	Security	Description	
PUBLISH_END	0x184		Publish configuration for event END	
PUBLISH_ENDRX	0x190		Publish configuration for event ENDRX	
PUBLISH_ACQUIRED	0x1A8		Publish configuration for event ACQUIRED	
SHORTS	0x200		Shortcuts between local events and tasks	
INTENSET	0x304		Enable interrupt	
INTENCLR	0x308		Disable interrupt	
SEMSTAT	0x400		Semaphore status register	
STATUS	0x440		Status from last transaction	
ENABLE	0x500		Enable SPI slave	
PSEL.SCK	0x508		Pin select for SCK	
PSEL.MISO	0x50C		Pin select for MISO signal	
PSEL.MOSI	0x510		Pin select for MOSI signal	
PSEL.CSN	0x514		Pin select for CSN signal	
PSELSCK	0x508		Pin select for SCK	Deprecated
PSELMISO	0x50C		Pin select for MISO	Deprecated
PSELMOSI	0x510		Pin select for MOSI	Deprecated
PSELCSN	0x514		Pin select for CSN	Deprecated
RXDPTR	0x534		RXD data pointer	Deprecated
MAXRX	0x538		Maximum number of bytes in receive buffer	Deprecated
AMOUNTRX	0x53C		Number of bytes received in last granted transaction	Deprecated
RXD.PTR	0x534		RXD data pointer	
RXD.MAXCNT	0x538		Maximum number of bytes in receive buffer	
RXD.AMOUNT	0x53C		Number of bytes received in last granted transaction	
RXD.LIST	0x540		EasyDMA list type	
TXDPTR	0x544		TXD data pointer	Deprecated
MAXTX	0x548		Maximum number of bytes in transmit buffer	Deprecated
AMOUNTTX	0x54C		Number of bytes transmitted in last granted transaction	Deprecated
TXD.PTR	0x544		TXD data pointer	
TXD.MAXCNT	0x548		Maximum number of bytes in transmit buffer	
TXD.AMOUNT	0x54C		Number of bytes transmitted in last granted transaction	
TXD.LIST	0x550		EasyDMA list type	
CONFIG	0x554		Configuration register	
DEF	0x55C		Default character. Character clocked out in case of an ignored transaction.	
ORC	0x5C0		Over-read character	

Table 78: Register overview

6.14.6.1 TASKS_ACQUIRE

Address offset: 0x024

Acquire SPI semaphore

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_ACQUIRE			Acquire SPI semaphore
			Trigger	1	Trigger task

6.14.6.2 TASKS_RELEASE

Address offset: 0x028

Release SPI semaphore, enabling the SPI slave to acquire it

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_RELEASE			Release SPI semaphore, enabling the SPI slave to acquire it
			Trigger	1	Trigger task

6.14.6.3 SUBSCRIBE_ACQUIRE

Address offset: 0x0A4

Subscribe configuration for task ACQUIRE

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task ACQUIRE will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.14.6.4 SUBSCRIBE_RELEASE

Address offset: 0x0A8

Subscribe configuration for task RELEASE

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task RELEASE will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.14.6.5 EVENTS_END

Address offset: 0x104

Granted transaction completed

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_END			Granted transaction completed
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.14.6.6 EVENTS_ENDRX

Address offset: 0x110

End of RXD buffer reached

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_ENDRX			End of RXD buffer reached
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.14.6.7 EVENTS_ACQUIRED

Address offset: 0x128

Semaphore acquired

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_ACQUIRED			Semaphore acquired
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.14.6.8 PUBLISH_END

Address offset: 0x184

Publish configuration for event END

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event END will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.14.6.9 PUBLISH_ENDRX

Address offset: 0x190

Publish configuration for event ENDRX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event ENDRX will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.14.6.10 PUBLISH_ACQUIRED

Address offset: 0x1A8

Publish configuration for event ACQUIRED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event ACQUIRED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.14.6.11 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

Bit number	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
ID																														A		
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	END_ACQUIRE			Shortcut between event END and task ACQUIRE
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut

6.14.6.12 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID																						C						B			A	
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	END			Write '1' to enable interrupt for event END
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	ENDRX			Write '1' to enable interrupt for event ENDRX
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	ACQUIRED			Write '1' to enable interrupt for event ACQUIRED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.14.6.13 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID																						C						B			A	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	END			Write '1' to disable interrupt for event END
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	ENDRX			Write '1' to disable interrupt for event ENDRX
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	ACQUIRED			Write '1' to disable interrupt for event ACQUIRED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.14.6.14 SEMSTAT

Address offset: 0x400

Semaphore status register

Bit number	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
ID																															A	A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	SEMSTAT			Semaphore status
			Free	0	Semaphore is free
			CPU	1	Semaphore is assigned to CPU
			SPIS	2	Semaphore is assigned to SPI slave
			CPUPending	3	Semaphore is assigned to SPI but a handover to the CPU is pending

6.14.6.15 STATUS

Address offset: 0x440

Status from last transaction

Individual bits are cleared by writing a '1' to the bits that shall be cleared

Bit number	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
ID																															B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	OVERREAD			TX buffer over-read detected, and prevented
			NotPresent	0	Read: error not present
			Present	1	Read: error present
			Clear	1	Write: clear error on writing '1'
B	RW	OVERFLOW			RX buffer overflow detected, and prevented
			NotPresent	0	Read: error not present
			Present	1	Read: error present
			Clear	1	Write: clear error on writing '1'

6.14.6.16 ENABLE

Address offset: 0x500

Enable SPI slave

Bit number	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
ID																													A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ENABLE			Enable or disable SPI slave
			Disabled	0	Disable SPI slave
			Enabled	2	Enable SPI slave

6.14.6.17 PSEL.SCK

Address offset: 0x508

Pin select for SCK

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.14.6.18 PSEL.MISO

Address offset: 0x50C

Pin select for MISO signal

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.14.6.19 PSEL.MOSI

Address offset: 0x510

Pin select for MOSI signal

Bit number	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
ID	C . . . . A A A A A
Reset 0xFFFFFFFF	**1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.14.6.20 PSEL.CSN

Address offset: 0x514

Pin select for CSN signal

Bit number	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
ID	C . . . . A A A A A
Reset 0xFFFFFFFF	**1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.14.6.21 PSELSCK (Deprecated)

Address offset: 0x508

Pin select for SCK

Bit number	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
ID	A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
Reset 0xFFFFFFFF	**1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PSELSCK		[0..31]	Pin number configuration for SPI SCK signal
			Disconnected	0xFFFFFFFF	Disconnect

6.14.6.22 PSELMISO (Deprecated)

Address offset: 0x50C

Pin select for MISO

Bit number	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
ID	A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
Reset 0xFFFFFFFF	**1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PSELMISO		[0..31]	Pin number configuration for SPI MISO signal
			Disconnected	0xFFFFFFFF	Disconnect

6.14.6.23 PSELMOSI (Deprecated)

Address offset: 0x510

Pin select for MOSI

Bit number	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
ID	A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
Reset 0xFFFFFFFF	**1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PSELMOSI		[0..31]	Pin number configuration for SPI MOSI signal
			Disconnected	0xFFFFFFFF	Disconnect

6.14.6.24 PSELCSN (Deprecated)

Address offset: 0x514

Pin select for CSN

Bit number	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
ID	A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
Reset 0xFFFFFFFF	**1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PSELCSN		[0..31]	Pin number configuration for SPI CSN signal
			Disconnected	0xFFFFFFFF	Disconnect

6.14.6.25 RXDPTR (Deprecated)

Address offset: 0x534

RXD data pointer

Bit number	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
ID	A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	RXDPTR			RXD data pointer

Note: See the memory chapter for details about which memories are available for EasyDMA.

6.14.6.26 MAXRX (Deprecated)

Address offset: 0x538

Maximum number of bytes in receive buffer

Bit number	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
ID	A A A A A A A A A A A A A (bits 12–0)
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXRX		[1..0x1FFF]	Maximum number of bytes in receive buffer

6.14.6.27 AMOUNTRX (Deprecated)

Address offset: 0x53C

Number of bytes received in last granted transaction

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	AMOUNTRX		[1..0x1FFF]	Number of bytes received in the last granted transaction

6.14.6.28 RXD.PTR

Address offset: 0x534

RXD data pointer

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			RXD data pointer

> **Note:** See the memory chapter for details about which memories are available for EasyDMA.

6.14.6.29 RXD.MAXCNT

Address offset: 0x538

Maximum number of bytes in receive buffer

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXCNT		[1..0x1FFF]	Maximum number of bytes in receive buffer

6.14.6.30 RXD.AMOUNT

Address offset: 0x53C

Number of bytes received in last granted transaction

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	AMOUNT		[1..0x1FFF]	Number of bytes received in the last granted transaction

6.14.6.31 RXD.LIST

Address offset: 0x540

EasyDMA list type

Bit number	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
ID																															A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LIST			List type
			Disabled	0	Disable EasyDMA list
			ArrayList	1	Use array list

6.14.6.32 TXDPTR (Deprecated)

Address offset: 0x544

TXD data pointer

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	TXDPTR			TXD data pointer

> **Note:** See the memory chapter for details about which memories are available for EasyDMA.

6.14.6.33 MAXTX (Deprecated)

Address offset: 0x548

Maximum number of bytes in transmit buffer

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXTX		[1..0x1FFF]	Maximum number of bytes in transmit buffer

6.14.6.34 AMOUNTTX (Deprecated)

Address offset: 0x54C

Number of bytes transmitted in last granted transaction

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	AMOUNTTX		[1..0x1FFF]	Number of bytes transmitted in last granted transaction

6.14.6.35 TXD.PTR

Address offset: 0x544

TXD data pointer

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			TXD data pointer

Note: See the memory chapter for details about which memories are available for EasyDMA.

6.14.6.36 TXD.MAXCNT

Address offset: 0x548

Maximum number of bytes in transmit buffer

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXCNT		[1..0x1FFF]	Maximum number of bytes in transmit buffer

6.14.6.37 TXD.AMOUNT

Address offset: 0x54C

Number of bytes transmitted in last granted transaction

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	AMOUNT		[1..0x1FFF]	Number of bytes transmitted in last granted transaction

6.14.6.38 TXD.LIST

Address offset: 0x550

EasyDMA list type

Bit number	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
ID																															A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LIST			List type
			Disabled	0	Disable EasyDMA list
			ArrayList	1	Use array list

6.14.6.39 CONFIG

Address offset: 0x554

Configuration register

Bit number	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
ID																														C	B	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	ORDER			Bit order
			MsbFirst	0	Most significant bit shifted out first
			LsbFirst	1	Least significant bit shifted out first
B	RW	CPHA			Serial clock (SCK) phase
			Leading	0	Sample on leading edge of clock, shift serial data on trailing edge
			Trailing	1	Sample on trailing edge of clock, shift serial data on leading edge
C	RW	CPOL			Serial clock (SCK) polarity
			ActiveHigh	0	Active high
			ActiveLow	1	Active low

6.14.6.40 DEF

Address offset: 0x55C

Default character. Character clocked out in case of an ignored transaction.

Bit number	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
ID																									A	A	A	A	A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	DEF			Default character. Character clocked out in case of an ignored transaction.

6.14.6.41 ORC

Address offset: 0x5C0

Over-read character

Bit number	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
ID																									A	A	A	A	A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	ORC			Over-read character. Character clocked out after an over-read of the transmit buffer.

6.14.7 Electrical specification

6.14.7.1 SPIS slave interface electrical specifications

Symbol	Description	Min.	Typ.	Max.	Units
f_{SPIS}	Bit rates for SPIS[11]			8[12]	Mbps
$t_{SPIS,START}$	Time from RELEASE task to receive/transmit (CSN active)		0.125		µs

11 High bit rates may require GPIOs to be set as High Drive, see GPIO chapter for more details.
12 The actual maximum data rate depends on the master's CLK to MISO and MOSI setup and hold timings.

6.14.7.2 Serial Peripheral Interface Slave (SPIS) timing specifications

Symbol	Description	Min.	Typ.	Max.	Units
$t_{SPIS,CSCKIN}$	SCK input period	125			ns
$t_{SPIS,RFSCKIN}$	SCK input rise/fall time			30	ns
$t_{SPIS,WHSCKIN}$	SCK input high time	30			ns
$t_{SPIS,WLSCKIN}$	SCK input low time	30			ns
$t_{SPIS,SUCSN}$	CSN to CLK setup time	1000			ns
$t_{SPIS,HCSN}$	CLK to CSN hold time	2000			ns
$t_{SPIS,ASA}$	CSN to MISO driven	0			ns
$t_{SPIS,ASO}$	CSN to MISO valid[a]			1000	ns
$t_{SPIS,DISSO}$	CSN to MISO disabled[a]			68	ns
$t_{SPIS,CWH}$	CSN inactive time	300			ns
$t_{SPIS,VSO}$	CLK edge to MISO valid			19	ns
$t_{SPIS,HSO}$	MISO hold time after CLK edge	18[13]			ns
$t_{SPIS,SUSI}$	MOSI to CLK edge setup time	59			ns
$t_{SPIS,HSI}$	CLK edge to MOSI hold time	20			ns

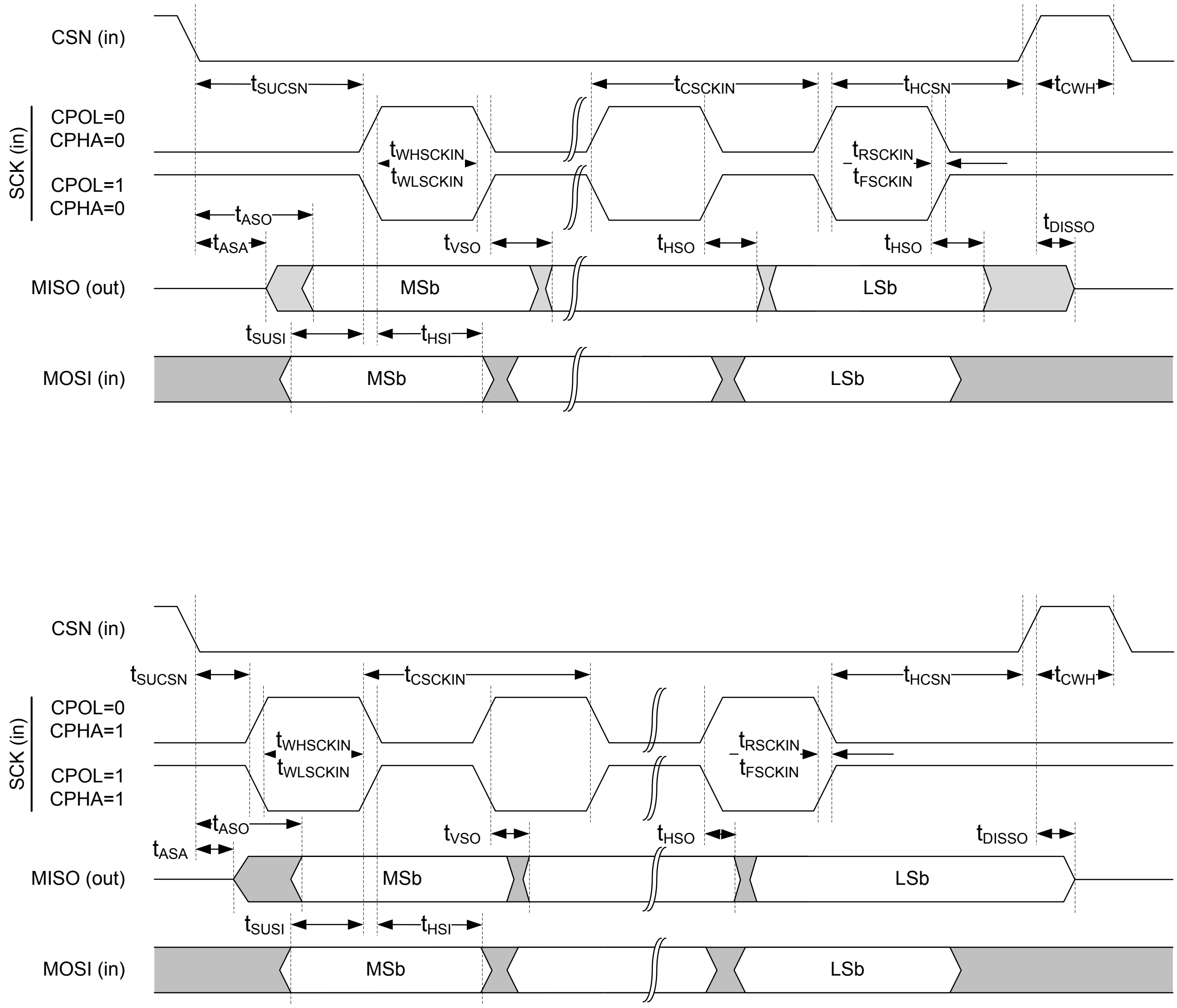

Figure 77: SPIS timing diagram

a At 25pF load, including GPIO capacitance, see GPIO spec.

13 This is to ensure compatibility to SPI masters sampling MISO on the same edge as MOSI is output

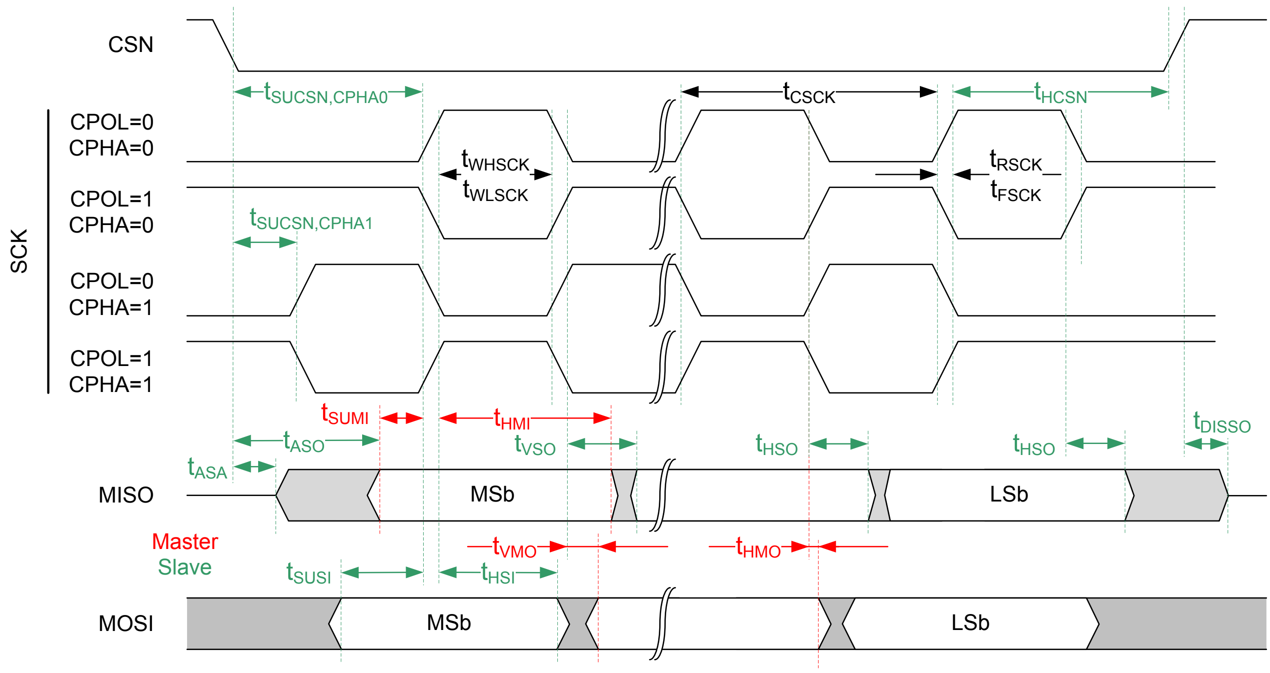

Figure 78: Common SPIM and SPIS timing diagram

6.15 SPU - System protection unit

SPU is the central point in the system to control access to memories, peripherals and other resources.

Listed here are the main features of the SPU:

- ARM® TrustZone® support, allowing definition of secure, non-secure and non-secure callable memory regions
- Extended ARM®TrustZone®, protecting memory regions and peripherals from non-CPU devices like EasyDMA transfer
- Pin access protection, preventing non-secure code and peripherals from accessing secure pin resources
- DPPI access protection, realized by preventing non-secure code and peripherals to publish from or subscribe to secured DPPI channels
- External domain access protection, controlling access rights from other MCUs

6.15.1 General concepts

SPU provides a register interface to control the various internal logic blocks that monitor access to memory-mapped slave devices (RAM, flash, peripherals, etc) and other resources (device pins, DPPI channels, etc).

For memory-mapped devices like RAM, flash and peripherals, the internal logic checks the address and attributes (e.g. read, write, execute, secure) of the incoming transfer to block it if necessary. Whether a secure resource can be accessed by a given master is defined:

For a CPU-type master
: By the security state of the CPU and the security state reported by the SPU, for the address in the bus transfer

For a non-CPU master
: By the security attribute of the master that initiates the transfer, defined by a SPU register

The Simplified view of the protection of RAM, flash and peripherals using SPU on page 261 shows a simplified view of the SPU registers controlling several internal modules.

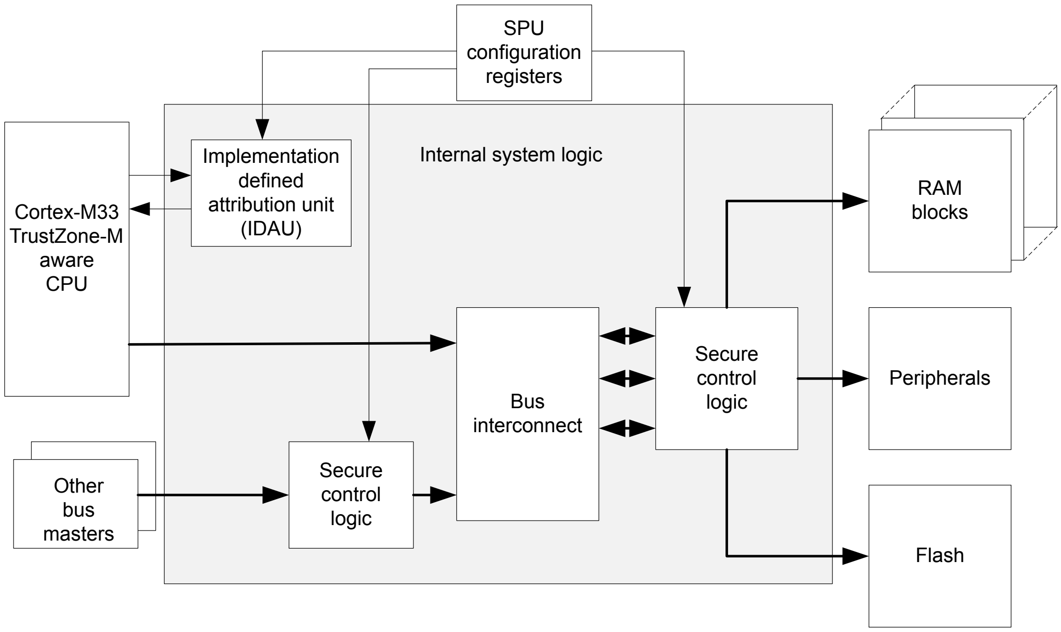

Figure 79: Simplified view of the protection of RAM, flash and peripherals using SPU

The protection logic implements a read-as-zero/write-ignore (RAZ/WI) policy:

- A blocked read operation will always return a zero value on the bus, preventing information leak
- A write operation to a forbidden region or peripheral will be ignored

An error is reported through dedicated error signals. For security state violations from an M33 master this will be a SecureFault exception, for other violations this will be an SPU event. The SPU event can be configured to generate an interrupt towards the CPU.

Other resources like pins and DPPI channels are protected by comparing the security attributes of the protected resource with the security attribute of the peripheral that wants to access it. The SPU is the only place where those security attributes can be configured.

6.15.1.1 Special considerations for ARM TrustZone for Cortex-M enabled system

For a ARM® TrustZone® for Cortex®-M enabled CPU, the SPU also controls custom logic.

Custom logic is shown as the implementation defined attribution unit (IDAU) in figure Simplified view of the protection of RAM, flash and peripherals using SPU on page 261. Full support is provided for:

- ARM® TrustZone® for Cortex®-M related instructions, like test target (TT) for reporting the security attributes of a region
- Non-secure callable (NSC) regions, to implement secure entry points from non-secure code

The SPU provides the necessary registers to configure the security attributes for memory regions and peripherals. However, as a requirement to use the SPU, the secure attribution unit (SAU) needs to be disabled and all memory set as non-secure in the ARM core. This will allow the SPU to control the IDAU and set the security attribution of all addresses as originally intended.

6.15.2 Flash access control

The flash memory space is divided into 32 regions of 32 KiB. For each region, four different types of permissions can be configured.

The four types of permissions are:

Read

Allows data read access to the region. Note that code fetch from this region is not controlled by the read permission but by the execute permission described below.

Write
Allows write or erase access to the region

Execute
Allows code fetch from this region, even if data read is disabled

Secure
Allows only bus accesses with the security attribute set to access the region

Permissions can be set independently. For example, it is possible to configure a flash region to be accessible only through secure transfer, being read-only (no write allowed) and non-executable (no code fetch allowed). For each region, permissions can be set and then locked by using the FLASHPERM[].PERM.LOCK bit, to prevent subsequent modifications.

Note that the debugger is able to step through execute-protected memory regions.

The following figure shows the flash memory space and the divided regions:

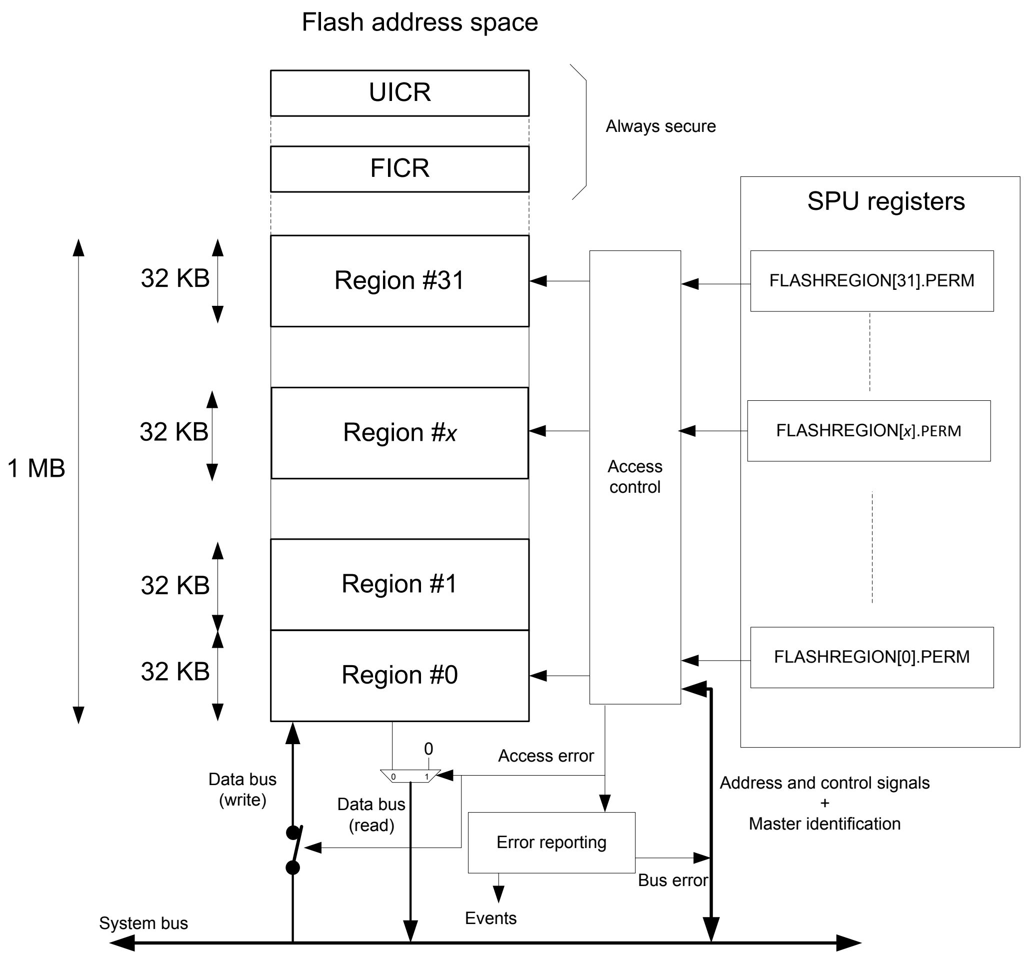

Figure 80: Region definition in the flash memory space

6.15.2.1 Non-secure callable (NSC) region definition in flash

The SPU provides support for the definition of non-secure callable (NSC) sub-regions to allow non-secure to secure function calls.

A non-secure callable sub-region can only exist within an existing secure region and its definition is done using two registers:

- FLASHNSC[].REGION, used to select the secure region that will contain the NSC sub-region
- FLASHNSC[].SIZE, used to define the size of the NSC sub-region within the secure region

The NSC sub-region will be defined from the highest address in that region, going downwards. Figure below illustrates the NSC sub-regions and the registers used for their definition:

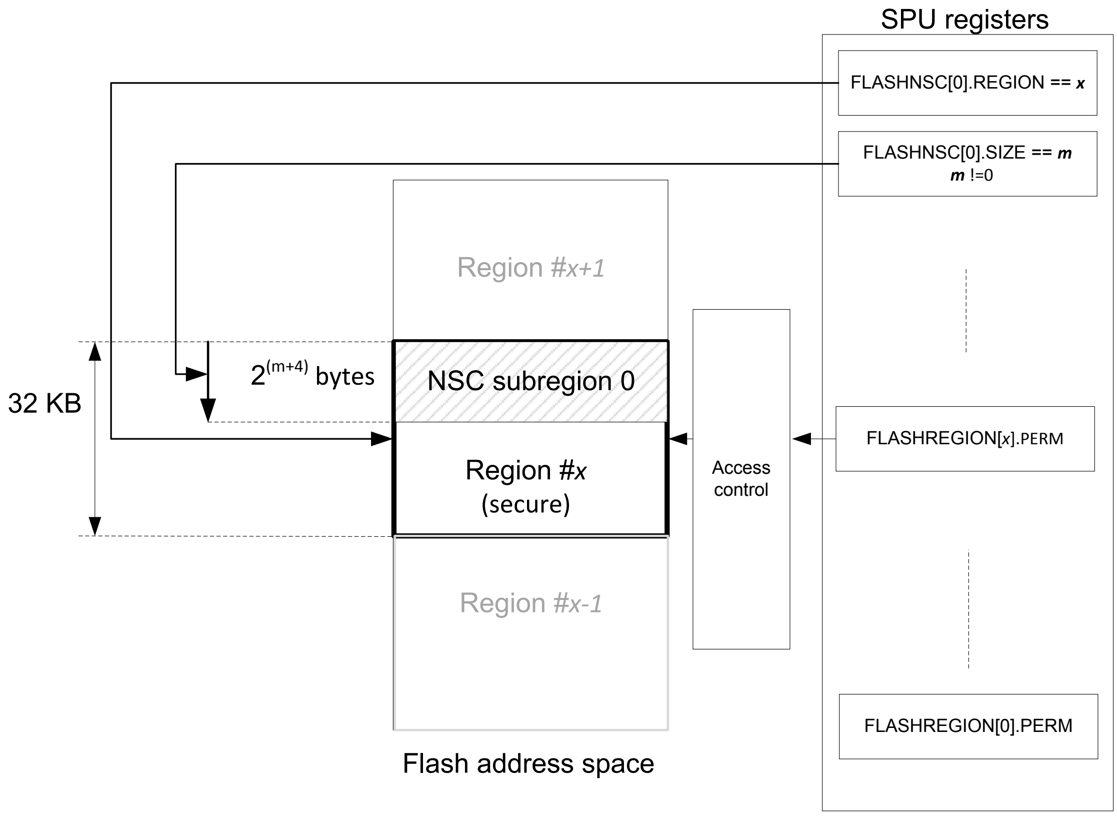

Figure 81: Non-secure callable region definition in the flash memory space

The NSC sub-region will only be defined if:

- FLASHNSC[*i*].SIZE value is non zero
- FLASHNSC[*i*].REGION defines a secure region

If FLASHNSC[*i*].REGION and FLASHNSC[*j*].REGION have the same value, there is only one sub-region defined as NSC, with the size given by the maximum of FLASHNSC[*i*].SIZE and FLASHNSC[*j*].SIZE.

If FLASHNSC[*i*].REGION defines a non-secure region, then there is no non-secure callable region defined and the selected region stays non-secure.

6.15.2.2 Flash access error reporting

The SPU and the logic controlled by it will respond with a certain behavior once an access violation is detected.

The following will happen once the logic controlled by the SPU detects an access violation on one of the flash ports:

- The faulty transfer will be blocked
- In case of a read transfer, the bus will be driven to zero
- Feedback will be sent to the master through specific bus error signals, if this is supported by the master. Moreover, the SPU will receive an event that can optionally trigger an interrupt towards the CPU.

- SecureFault exception will be triggered if security violation is detected for access from Cortex®-M33
- BusFault exception will be triggered when read/write/execute protection violation is detected for Cortex®-M33
- FLASHACCERR event will be triggered if any access violations are detected for all master types except for Cortex®-M33 security violation

The following table summarizes the SPU behavior based on the type of initiator and access violation:

Master type	Security violation	Read/Write/Execute protection violation
Cortex®-M33	SecureFault exception	BusFault exception, FLASHACCERR event
EasyDMA	RAZ/WI, FLASHACCERR event	RAZ/WI, FLASHACCERR event
Other masters	RAZ/WI, FLASHACCERR event	RAZ/WI, FLASHACCERR event

Table 79: Error reporting for flash access errors

For a Cortex®-M33 master, the SecureFault exception will take precedence over the BusFault exception if a security violation occurs simultaneously with another type of violation.

6.15.2.3 UICR and FICR protections

The user information configuration registers (UICR) and factory information configuration registers (FICR) are always considered as secure. FICR registers are read-only. UICR registers can be read and written by secure code only.

Writing new values to user information configuration registers must follow the procedure described in NVMC — Non-volatile memory controller on page 28. Code execution from FICR and UICR address spaces will always be reported as access violation, an exception to this rule applies during a debug session.

6.15.3 RAM access control

Each RAM memory space region has a set of permissions that can be set independently.

The RAM memory space is divided into 32 regions of 8 KiB.

For each region, four different types of permissions can be configured:

Read
: Allows data read access to the region. Code fetch from this region is not controlled by the read permission but by the execute permission described below.

Write
: Allows write access to the region

Execute
: Allows code fetch from this region

Secure
: Allows only bus accesses with the security attribute set to access the region

Permissions can be set independently. For example, it is possible to configure a RAM region to be accessible only through secure transfer, being read-only (no write allowed) and non-executable (no code fetch allowed). For each region, permissions can be set and then locked to prevent subsequent modifications by using the RAMPERM[].PERM.LOCK bit.

The following figure shows the RAM memory space and the devided regions:

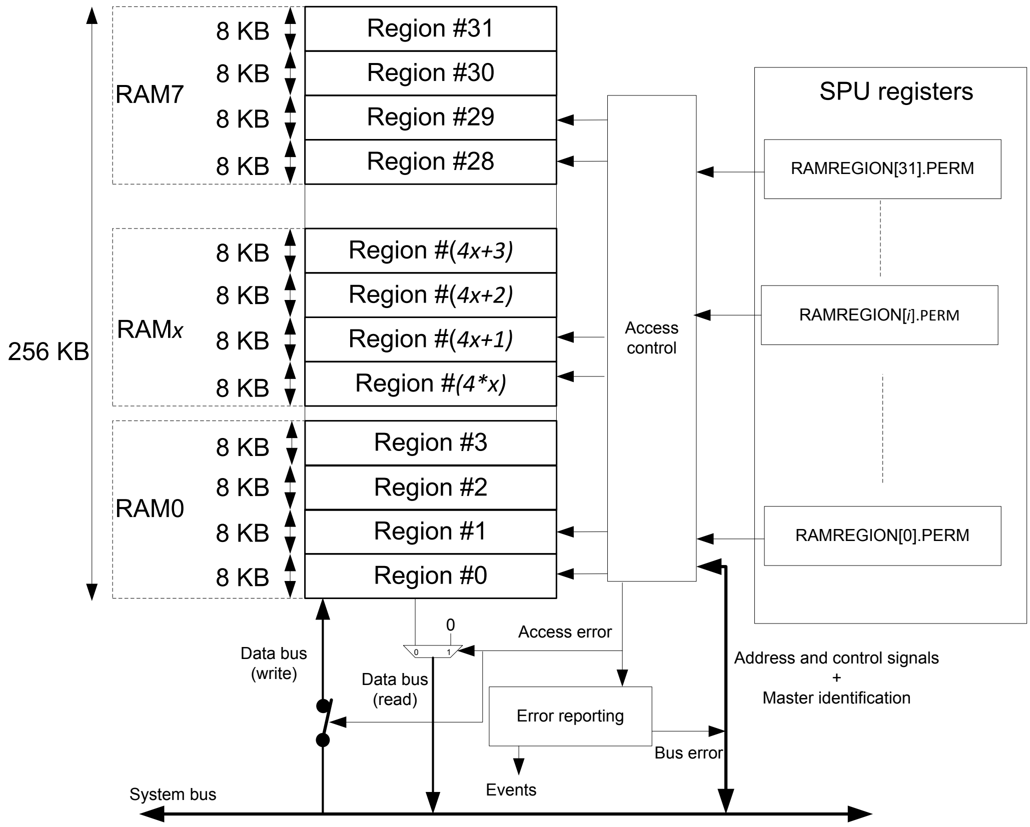

Figure 82: Region definition in the RAM memory space

6.15.3.1 Non-secure callable (NSC) region definition in RAM

The SPU provides support for the definition of non-secure callable (NSC) sub-regions to allow non-secure to secure function calls.

A non-secure callable sub-region can only exist within an existing secure region and its definition is done using two registers:

- RAMNSC[].REGION, used to select the secure region that will contain the NSC sub-region
- RAMNSC[].SIZE, used to define the size of the NSC sub-region within the secure region

The NSC sub-region will be defined from the highest address in that region, going downwards. Figure below illustrates the NSC sub-regions and the registers used for their definition:

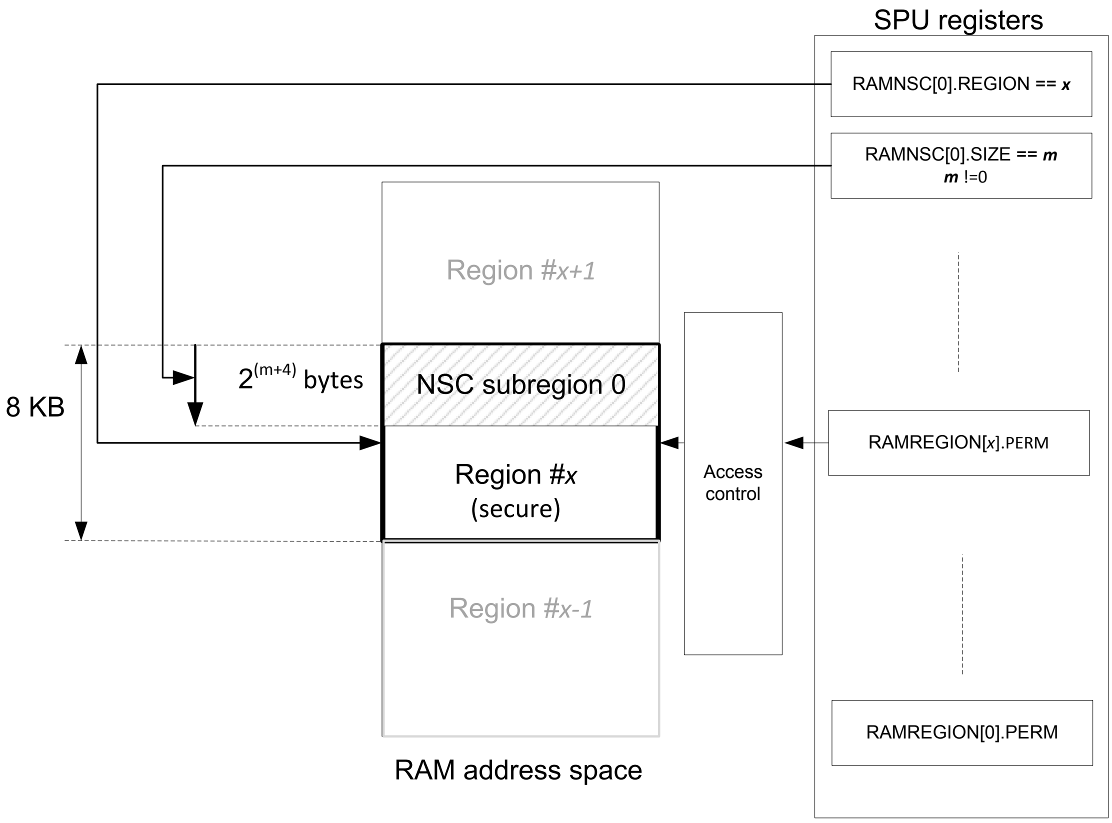

Figure 83: Non-secure callable region definition in the RAM memory space

The NSC sub-region will only be defined if:

- RAMNSC[*i*].SIZE value is non zero
- RAMNSC[*i*].REGION defines a secure region

If RAMNSC[*i*].REGION and RAMNSC[*j*].REGION have the same value, there is only one sub-region defined as NSC, with the size given by the maximum of RAMNSC[*i*].SIZE and RAMNSC[*j*].SIZE.

If RAMNSC[*i*].REGION defines a non-secure region, then there is no non-secure callable region defined and the selected region stays non-secure.

6.15.3.2 RAM access error reporting

The SPU and the logic controlled by it will respond with a certain behavior once an access violation is detected.

The following will happen once the logic controlled by the SPU detects an access violation on one of the RAM ports:

- The faulty transfer will be blocked
- In case of a read transfer, the bus will be driven to zero
- Feedback will be sent to the master through specific bus error signals, if this is supported by the master
- SecureFault exception will be triggered if security violation is detected for access from Cortex®-M33
- BusFault exception will be triggered when read/write/execute protection violation is detected for Cortex®-M33. The SPU will also generate an event that can optionally trigger an interrupt towards the CPU.
- RAMACCERR event will be triggered if any access violations are detected for all master types but for Cortex®-M33 security violation

The following table summarizes the SPU behavior based on the type of initiator and access violation:

Master type	Security violation	Read/Write/Execute protection violation
Cortex®-M33	SecureFault exception	BusFault exception, RAMACCERR event
EasyDMA	RAZ/WI, RAMACCERR event	RAZ/WI, RAMACCERR event
Other masters	RAZ/WI, RAMACCERR event	RAZ/WI, RAMACCERR event

Table 80: Error reporting for RAM access errors

For a Cortex®-M33 master, the SecureFault exception will take precedence over the BusFault exception if a security violation occurs simultaneously with another type of violation.

6.15.4 Peripheral access control

Access controls are defined by the characteristics of the peripheral.

Peripherals can have their security attribute set as:

Always secure
For a peripheral related to system control

Always non-secure
For some general-purpose peripherals

Configurable
For general-purpose peripherals that may be configured for secure only access

The full list of peripherals and their corresponding security attributes can be found in Memory map on page 22. For each peripheral with ID *id*, PERIPHID[*id*]. PERM will show whether the security attribute for this peripheral is configurable or not.

If not hardcoded, the security attribute can configured using the PERIPHID[*id*].PERM.

At reset, all user-selectable and split security peripherals are set to be secure, with secure DMA where present.

Secure code can access both secure peripherals and non-secure peripherals.

6.15.4.1 Peripherals with split security

Peripherals with split security are defined to handle use-cases when both secure and non-secure code needs to control the same resource.

When peripherals with split security have their security attribute set to non-secure, access to specific registers and bitfields within some registers is dependent on the security attribute of the bus transfer. For example, some registers will not be accessible for a non-secure transfer.

When peripherals with split security have their security attribute set to secure, then only secure transfers can access their registers.

See Instantiation on page 23 for an overview of split security peripherals. Respective peripheral chapters explain the specific security behavior of each peripheral.

6.15.4.2 Peripheral address mapping

Peripherals that have non-secure security mapping have their address starting with 0x4XXX_XXXX. Peripherals that have secure security mapping have their address starting with 0x5XXX_XXXX.

Peripherals with a user-selectable security mapping are available at an address starting with:

- 0x4XXX_XXXX, if the peripheral security attribute is set to non-secure
- 0x5XXX_XXXX, if the peripheral security attribute is set to secure

Peripherals with a split security mapping are available at an address starting with:

- 0x4XXX_XXXX for non-secure access and 0x5XXX_XXXX for secure access, if the peripheral security attribute is set to non-secure
    - Secure registers in the 0x4XXX_XXXX range are not visible for secure or non-secure code, and an attempt to access such a register will result in write-ignore, read-as-zero behavior
    - Secure code can access both non-secure and secure registers in the 0x5XXX_XXXX range
- 0x5XXX_XXXX, if the peripheral security attribute is set to secure

Any attempt to access the 0x5000_0000-0x5FFF_FFFF address range from non-secure code will be ignored and generate a SecureFault exception.

The table below illustrates the address mapping for the three type of peripherals in all possible configurations

Security-features and configuration	Is mapped at 0x4XXX_XXXX?	Is mapped at 0x5XXX_XXXX?
Secure peripheral	No	Yes
Non-secure peripheral	Yes	No
Split-security peripheral, with attribute=secure	No	Yes
Split-security peripheral, with attribute=non-secure	Yes, restricted functionality	Yes

Table 81: Peripheral's address mapping in relation to its security-features and configuration

6.15.4.3 Special considerations for peripherals with DMA master

Peripherals containing a DMA master can be configured so the security attribute of the DMA transfers is different from the security attribute of the peripheral itself. This allows a secure peripheral to do non-secure data transfers to or from the system memories.

The following conditions must be met:

- The DMA field of PERIPHID[].PERM.SECURITYMAPPING should read as "SeparateAttribute"
- The peripheral itself should be secure (PERIPHID[].PERM.SECATTR == 1)

Then it is possible to select the security attribute of the DMA transfers using the field DMASEC (PERIPHID[].PERM.DMASEC == Secure and PERIPHID[].PERM.DMASEC == NonSecure) in PERIPHID[].PERM.

6.15.4.4 Peripheral access error reporting

Peripherals send error reports once access violation is detected.

The following will happen if the logic controlled by the SPU detects an access violation on one of the peripherals:

- The faulty transfer will be blocked
- In case of a read transfer, the bus will be driven to zero
- Feedback is sent to the master through specific bus error signals, if this is supported by the master. If the master is a processor supporting ARM® TrustZone® for Cortex®-M, a SecureFault exception will be generated for security related errors.
- The PERIPHACCERR event will be triggered

6.15.5 Pin access control

Access to device pins can be controlled by the SPU. A pin can be declared as secure so that only secure peripherals or secure code can access it.

The security attribute of each pin can be individually configured in SPU's GPIOPORT[0].PERM register. When the secure attribute is set for a pin, only peripherals that have the secure attribute set will be able to read the value of the pin or change it.

Peripherals can select the pin(s) they need access to through their PSEL register(s). If a peripheral has its attribute set to non-secure, but one of its PSEL registers selects a pin with the attribute set to secure, the SPU controlled logic will ensure that the pin selection is not propagated. In addition, the pin value will always be read as zero, to prevent a non-secure peripheral from obtaining a value from a secure pin. Whereas access to other pins with attribute set as non-secure will not be blocked.

Peripherals located in other domains (other than the application domain) can access pins only if the security attribute of the domain allows access to the pins they are trying to access. That is, secure domains can access both secure and non-secure pins, whereas non-secure domains can only access non-secure pins. This is illustrated in the following figure:

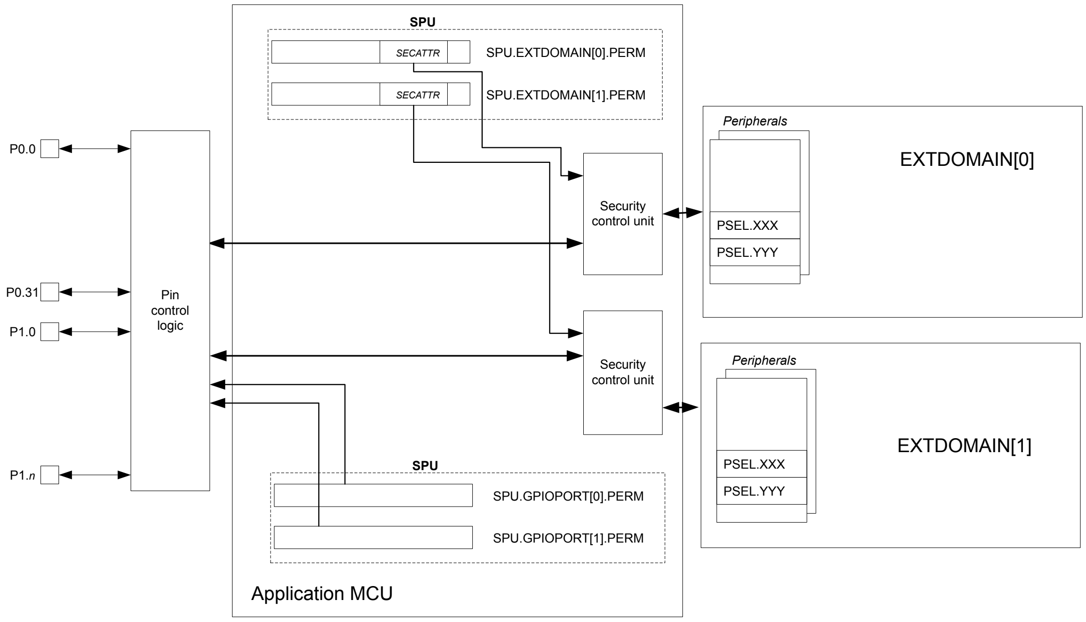

Figure 84: Pin access for domains other than the application domain

6.15.5.1 Direct pin control through the GPIO peripheral

Pins can be controlled directly through the general purpose input/output (GPIO) peripheral. It is a split-security peripheral, meaning that both secure and non-secure accesses are allowed.

Non-secure peripheral accesses will only be able to configure and control pins defined as non-secure in the GPIOPORT.PERM[] register(s). Attempts to access a register or a bitfield controlling a pin marked as secure in GPIO.PERM[] register(s) will be ignored:

- Write accesses will have no effect
- Read accesses will always return a zero value

No exception is triggered when a non-secure access targets a register or bitfield controlling a secure pin. For example, if the bit *i* is set in the GPIO.PERM[0] register (declaring Pin P0.*i* as secure), then

- non-secure write accesses to OUT, OUTSET, OUTCLR, DIR, DIRSET, DIRCLR and LATCH registers will not be able to write to bit *i* of those registers
- non-secure write accesses to register PIN_CNF[*i*] will be ignored
- non-secure read accesses to registers OUT, OUTSET, OUTCLR, IN, DIR, DIRSET, DIRCLR and LATCH will always read a 0 for the bit at position *i*
- non-secure read accesses to register PIN_CNF[*i*] will always return 0

The GPIO.DETECTMODE and GPIO.DETECTMODE_SEC registers are handled differently than the other registers mentioned before. When securely accessed, the DETECTMODE_SEC register controls the source for the DETECT_SEC signal for the pins marked as secure. Upon a non-secure access, the DETECTMODE_SEC is read as zero and write access are ignored. The GPIO.DETECTMODE register controls the source for the DETECT_NSEC signal for the pins defined as non-secured.

The DETECT_NSEC signal is routed to the non-secure GPIOTE peripheral, GPIOTE1, allowing generation of events and interrupts from pins marked as non-secured. The DETECT_SEC signal is routed to the secure GPIOTE peripheral, GPIOTE0, allowing generation of events and interrupts from pins marked as secured. The following figure illustrates how the DETECT_NSEC and DETECT_SEC signals are generated from the GPIO PIN[].DETECT signals.

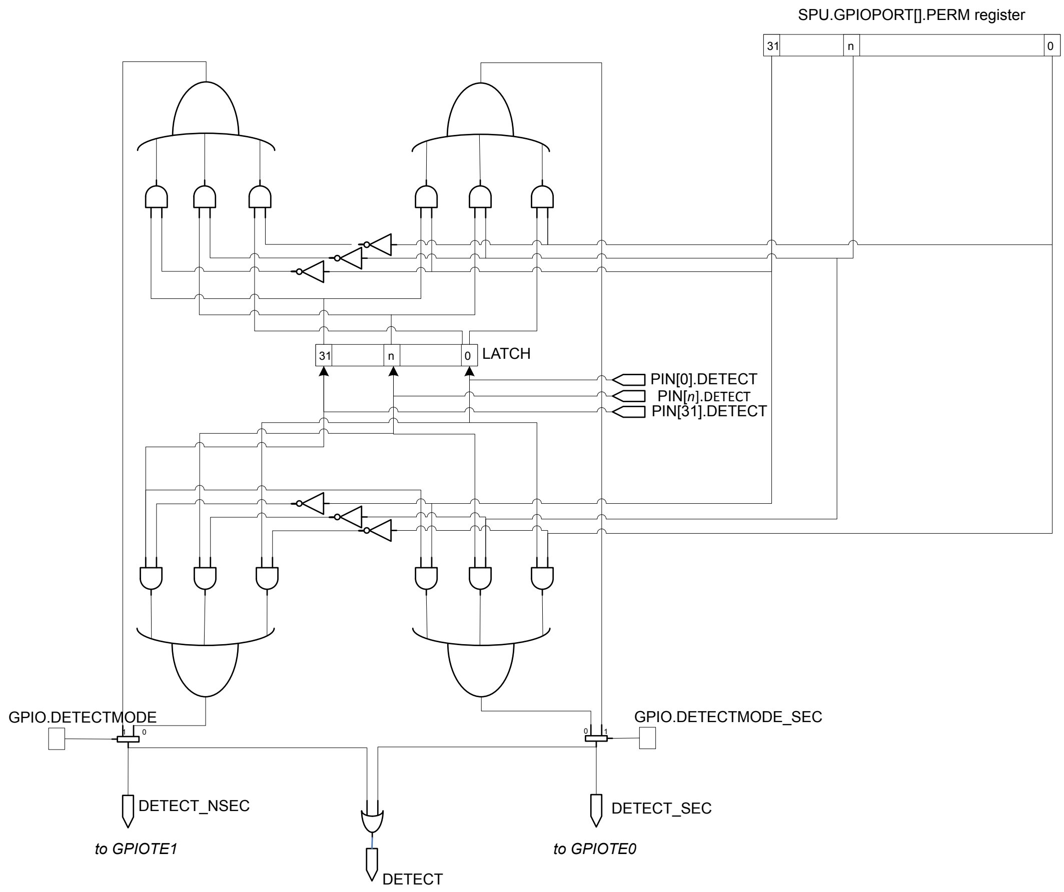

Figure 85: Principle of direct pin access

6.15.6 DPPI access control

Access to DPPI channels can be restricted. A channel can be declared as secure so that only secure peripherals can access it.

The security attribute of a DPPI channel is configured in . When the secure attribute is set for a channel, only peripherals that have the secure attribute set will be able to publish events to this channel or subscribe to this channel to receive tasks.

The DPPI controller peripheral (DPPIC) is a split security peripheral, i.e., its security behavior depends on the security attributes of both the DPPIC and the accessing party. See Special considerations regarding the DPPIC configuration registers on page 271 for more information about the DPPIC security behavior.

If a non-secure peripheral wants to publish an event on a secure DPPI channel, the channel will ignore the event. If a non-secure peripheral subscribes to a secure DPPI channel, it will not receive any events from this channel. The following figure illustrates the principle of operation of the security logic for a subscribed channel:

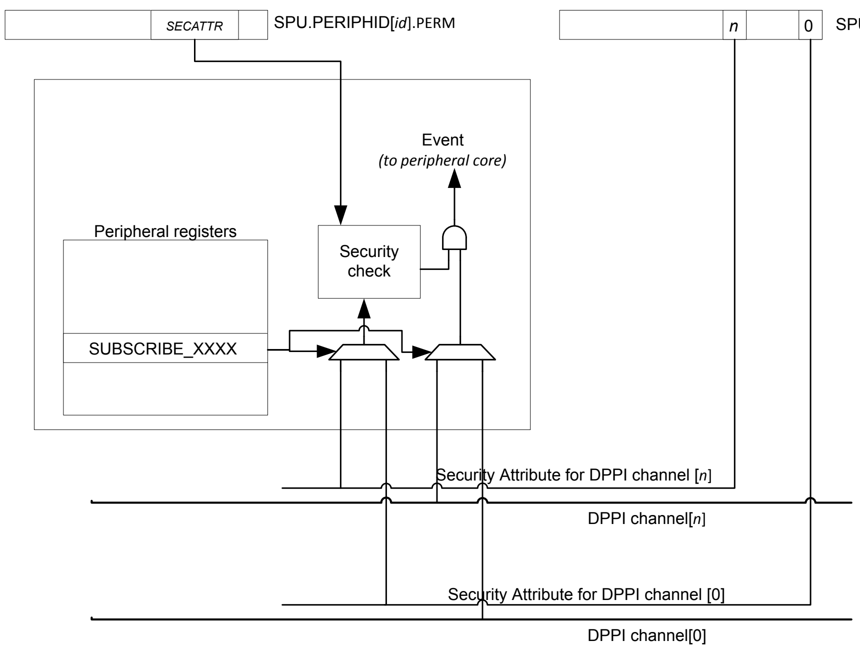

Figure 86: Subscribed channel security concept

No error reporting mechanism is associated with the DPPI access control logic.

6.15.6.1 Special considerations regarding the DPPIC configuration registers

DPPI channels can be enabled, disabled and grouped through the DPPI controller (DPPIC). The DPPIC is a split-security peripheral, and handles both secure and non-secure accesses.

A non-secure peripheral access will only be able to configure and control DPPI channels defined as non-secure in SPU's DPPI[n].PERM register(s). A secure peripheral access can control all DPPI channels, independently of the configuration in the DPPI[n].PERM register(s).

The DPPIC allows the creation of group of channels to be able to enable or disable all channels within a group simultaneously. The security attribute of a group of channels (secure or non-secure) is defined as follows:

- If all channels (enabled or not) in the group are non-secure, then the group is considered non-secure
- If at least one of the channels (enabled or not) in the group is secure, then the group is considered secure

A non-secure access to a DPPIC register, or a bitfield controlling a channel marked as secure in SPU.DDPI[n].PERM register(s), will be ignored:

- Write accesses will have no effect
- Read will always return a zero value

No exceptions are thrown when a non-secure access targets a register or bitfield controlling a secure channel. For example, if the bit *i* is set in the SPU.DPPI[0].PERM register (declaring the DPPI channel *i* as secure), then:

- Non-secure write accesses to registers CHEN, CHENSET and CHENCLR will not be able to write to bit *i* of those registers
- Non-secure write accesses to registers TASK_CHG[*j*].EN and TASK_CHG[*j*].DIS will be ignored if the channel group *j* contains at least one channel defined as secure (it can be the channel *i* itself or any channel declared as secured)

- Non-secure read accesses to registers CHEN, CHENSET and CHENCLR will always read zero for the bit at position *i*

For the channel configuration registers (DPPIC.CHG[...]), access from non-secure code is only possible if the included channels are all non-secure, whether the channels are enabled or not. If a DPPIC.CHG[*g*] register included one or more secure channels, then the group *g*is considered as secure and only a secure transfer can read or write DPPIC.CHG[*g*]. A non-secure write will be ignored and a non-secure read will return zero.

The DPPIC can subscribe to secure or non-secure channels through SUBSCRIBE_CHG[] registers in order to trigger task for enabling or disabling groups of channels. But an event from a non-secure channel will be ignored if the group subscribing to this channel is secure. An event from a secure channel can trigger both secure and non-secure tasks.

6.15.7 External domain access control

Other domains with their own CPUs can access peripherals, flash and RAM memories. The SPU allows controlling accesses from those bus masters.

The external domains can access application MCU memories and peripherals. External domains are assigned security attributes as described in register EXTDOMAIN[n].PERM.

Domain	Capability register	Permission register
LTE modem	Modem is always a non-secure domain	Not applicable

Table 82: Register mapping for external domains

The figure below illustrates how the security control units are used to assign security attributes to transfers initiated by the external domains:

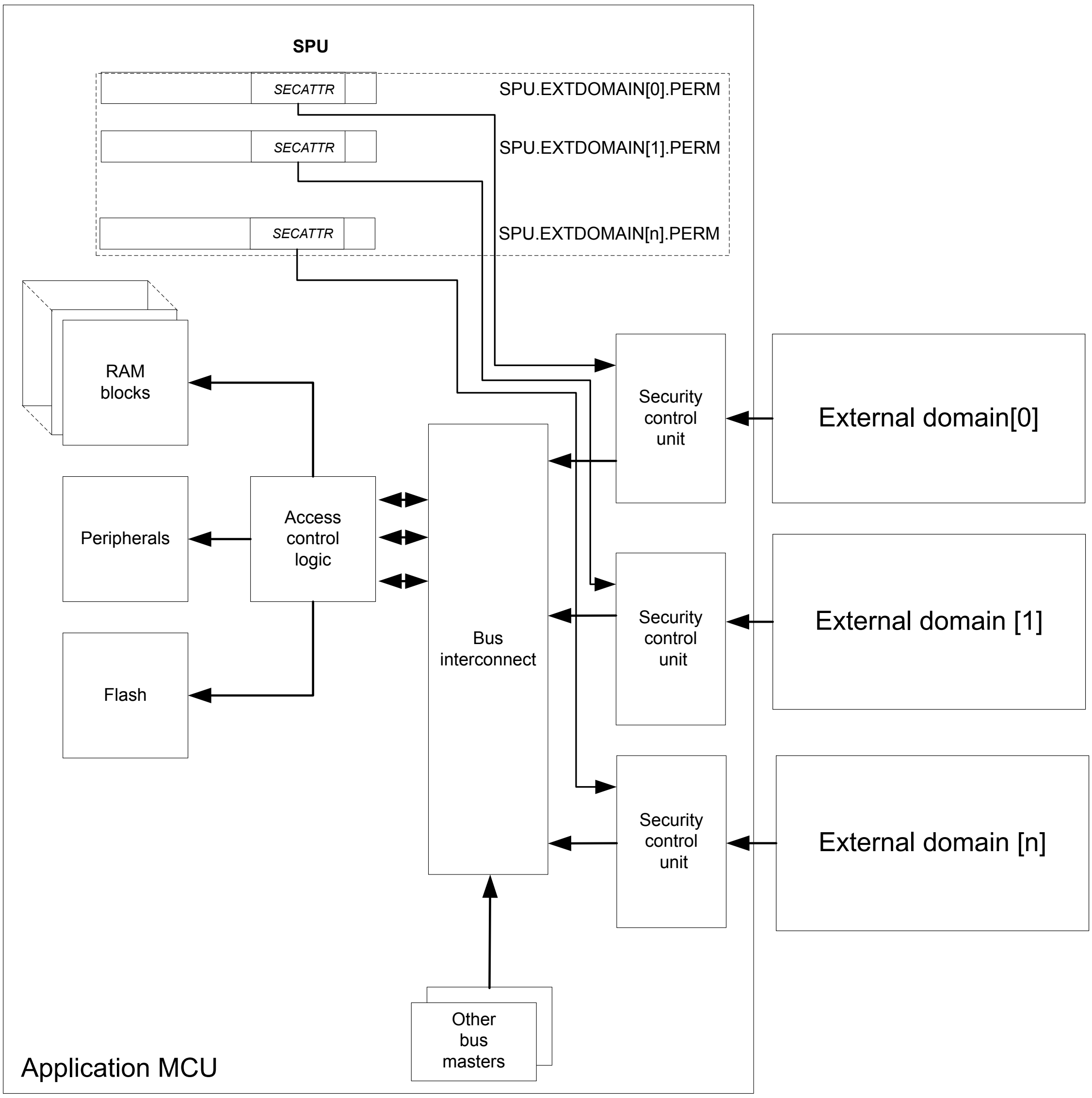

Figure 87: Access control from external domains

6.15.8 TrustZone for Cortex-M ID allocation

Flash and RAM regions, as well as non-secure and secure peripherals, are assigned unique TrustZone® IDs.

Note: TrustZone® ID should not be confounded with the peripheral ID used to identify peripherals.

The table below shows the TrustZone® ID allocation:

Regions	TrustZone Cortex-M ID
Flash regions 0..31	0..31
RAM regions 0..15	64..79
Non-secure peripherals	253
Secure peripherals	254

Table 83: TrustZone ID allocation

6.15.9 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50003000	SPU	SPU	S	NA	System Protection Unit	

Table 84: Instances

Register	Offset	Security	Description	
EVENTS_RAMACCERR	0x100		A security violation has been detected for the RAM memory space	
EVENTS_FLASHACCERR	0x104		A security violation has been detected for the flash memory space	
EVENTS_PERIPHACCERR	0x108		A security violation has been detected on one or several peripherals	
PUBLISH_RAMACCERR	0x180		Publish configuration for event RAMACCERR	
PUBLISH_FLASHACCERR	0x184		Publish configuration for event FLASHACCERR	
PUBLISH_PERIPHACCERR	0x188		Publish configuration for event PERIPHACCERR	
INTEN	0x300		Enable or disable interrupt	
INTENSET	0x304		Enable interrupt	
INTENCLR	0x308		Disable interrupt	
CAP	0x400		Show implemented features for the current device	
EXTDOMAIN[0].PERM	0x440		Access for bus access generated from the external domain 0 List capabilities of the external domain 0	
DPPI[0].PERM	0x480		Select between secure and non-secure attribute for the DPPI channels.	
DPPI[0].LOCK	0x484		Prevent further modification of the corresponding PERM register	
GPIOPORT[0].PERM	0x4C0		Select between secure and non-secure attribute for pins 0 to 31 of port 0.	Retained
GPIOPORT[0].LOCK	0x4C4		Prevent further modification of the corresponding PERM register	
FLASHNSC[0].REGION	0x500		Define which flash region can contain the non-secure callable (NSC) region 0	
FLASHNSC[0].SIZE	0x504		Define the size of the non-secure callable (NSC) region 0	
FLASHNSC[1].REGION	0x508		Define which flash region can contain the non-secure callable (NSC) region 1	
FLASHNSC[1].SIZE	0x50C		Define the size of the non-secure callable (NSC) region 1	
RAMNSC[0].REGION	0x540		Define which RAM region can contain the non-secure callable (NSC) region 0	
RAMNSC[0].SIZE	0x544		Define the size of the non-secure callable (NSC) region 0	
RAMNSC[1].REGION	0x548		Define which RAM region can contain the non-secure callable (NSC) region 1	
RAMNSC[1].SIZE	0x54C		Define the size of the non-secure callable (NSC) region 1	
FLASHREGION[n].PERM	0x600		Access permissions for flash region n	
RAMREGION[n].PERM	0x700		Access permissions for RAM region n	
PERIPHID[n].PERM	0x800		List capabilities and access permissions for the peripheral with ID n	

Table 85: Register overview

6.15.9.1 EVENTS_RAMACCERR

Address offset: 0x100

A security violation has been detected for the RAM memory space

Bit number	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
ID	A (bit 0)
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_RAMACCERR			A security violation has been detected for the RAM memory space
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.15.9.2 EVENTS_FLASHACCERR

Address offset: 0x104

A security violation has been detected for the flash memory space

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_FLASHACCERR			A security violation has been detected for the flash memory space
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.15.9.3 EVENTS_PERIPHACCERR

Address offset: 0x108

A security violation has been detected on one or several peripherals

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_PERIPHACCERR			A security violation has been detected on one or several peripherals
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.15.9.4 PUBLISH_RAMACCERR

Address offset: 0x180

Publish configuration for event RAMACCERR

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event RAMACCERR will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.15.9.5 PUBLISH_FLASHACCERR

Address offset: 0x184

Publish configuration for event FLASHACCERR

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event FLASHACCERR will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.15.9.6 PUBLISH_PERIPHACCERR

Address offset: 0x188

Publish configuration for event PERIPHACCERR

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event PERIPHACCERR will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.15.9.7 INTEN

Address offset: 0x300

Enable or disable interrupt

Bit number	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
ID																														C	B	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	RAMACCERR			Enable or disable interrupt for event RAMACCERR
			Disabled	0	Disable
			Enabled	1	Enable
B	RW	FLASHACCERR			Enable or disable interrupt for event FLASHACCERR
			Disabled	0	Disable
			Enabled	1	Enable
C	RW	PERIPHACCERR			Enable or disable interrupt for event PERIPHACCERR
			Disabled	0	Disable
			Enabled	1	Enable

6.15.9.8 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID																														C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	RAMACCERR			Write '1' to enable interrupt for event RAMACCERR
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	FLASHACCERR			Write '1' to enable interrupt for event FLASHACCERR
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	PERIPHACCERR			Write '1' to enable interrupt for event PERIPHACCERR
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.15.9.9 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID																														C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	RAMACCERR			Write '1' to disable interrupt for event RAMACCERR
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	FLASHACCERR			Write '1' to disable interrupt for event FLASHACCERR
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	PERIPHACCERR			Write '1' to disable interrupt for event PERIPHACCERR
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.15.9.10 CAP

Address offset: 0x400

Show implemented features for the current device

Bit number	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
ID																																A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	TZM			Show ARM TrustZone status
			NotAvailable	0	ARM TrustZone support not available
			Enabled	1	ARM TrustZone support is available

6.15.9.11 EXTDOMAIN[n].PERM (n=0..0)

Address offset: 0x440 + (n × 0x4)

Access for bus access generated from the external domain n

List capabilities of the external domain n

Bit number	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
ID																								C				B			A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	SECUREMAPPING			Define configuration capabilities for TrustZone Cortex-M secure attribute **Note:** This field is ReadOnly
			NonSecure	0	The bus access from this external domain always have the non-secure attribute set
			Secure	1	The bus access from this external domain always have the secure attribute set
			UserSelectable	2	Non-secure or secure attribute for bus access from this domain is defined by the EXTDOMAIN[n].PERM register
B	RW	SECATTR			Peripheral security mapping **Note:** This bit has effect only if EXTDOMAIN[n].PERM.SECUREMAPPING reads as UserSelectable
			NonSecure	0	Bus accesses from this domain have the non-secure attribute set
			Secure	1	Bus accesses from this domain have secure attribute set
C	RW	LOCK			
			Unlocked	0	This register can be updated
			Locked	1	The content of this register can't be changed until the next reset

6.15.9.12 DPPI[n].PERM (n=0..0)

Address offset: 0x480 + (n × 0x8)

Select between secure and non-secure attribute for the DPPI channels.

Bit number	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
ID																	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0x0000FFFF	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A-P	RW	CHANNEL[i] (i=0..15)			Select secure attribute.
			Secure	1	Channeli has its secure attribute set
			NonSecure	0	Channeli has its non-secure attribute set

6.15.9.13 DPPI[n].LOCK (n=0..0)

Address offset: 0x484 + (n × 0x8)

Prevent further modification of the corresponding PERM register

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LOCK			
			Locked	1	DPPI[n].PERM register can't be changed until next reset
			Unlocked	0	DPPI[n].PERM register content can be changed

6.15.9.14 GPIOPORT[n].PERM (n=0..0) (Retained)

Address offset: 0x4C0 + (n × 0x8)

This register is a retained register

Select between secure and non-secure attribute for pins 0 to 31 of port n.

Bit number	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
ID	f	e	d	c	b	a	Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A-f	RW	PIN[i] (i=0..31)			Select secure attribute attribute for PIN i.
			Secure	1	Pin i has its secure attribute set
			NonSecure	0	Pin i has its non-secure attribute set

6.15.9.15 GPIOPORT[n].LOCK (n=0..0)

Address offset: 0x4C4 + (n × 0x8)

Prevent further modification of the corresponding PERM register

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LOCK			
			Locked	1	GPIOPORT[n].PERM register can't be changed until next reset
			Unlocked	0	GPIOPORT[n].PERM register content can be changed

6.15.9.16 FLASHNSC[n].REGION (n=0..1)

Address offset: 0x500 + (n × 0x8)

Define which flash region can contain the non-secure callable (NSC) region n

Bit number	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
ID																								B				A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	REGION			Region number
B	RW	LOCK			
			Unlocked	0	This register can be updated
			Locked	1	The content of this register can't be changed until the next reset

6.15.9.17 FLASHNSC[n].SIZE (n=0..1)

Address offset: 0x504 + (n × 0x8)

Define the size of the non-secure callable (NSC) region n

Bit number	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
ID																								B					A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	SIZE			Size of the non-secure callable (NSC) region n
			Disabled	0	The region n is not defined as a non-secure callable region. Normal security attributes (secure or non-secure) are enforced.
			32	1	The region n is defined as non-secure callable with a 32-byte size
			64	2	The region n is defined as non-secure callable with a 64-byte size
			128	3	The region n is defined as non-secure callable with a 128-byte size
			256	4	The region n is defined as non-secure callable with a 256-byte size
			512	5	The region n is defined as non-secure callable with a 512-byte size
			1024	6	The region n is defined as non-secure callable with a 1024-byte size
			2048	7	The region n is defined as non-secure callable with a 2048-byte size
			4096	8	The region n is defined as non-secure callable with a 4096-byte size
B	RW	LOCK			
			Unlocked	0	This register can be updated
			Locked	1	The content of this register can't be changed until the next reset

6.15.9.18 RAMNSC[n].REGION (n=0..1)

Address offset: 0x540 + (n × 0x8)

Define which RAM region can contain the non-secure callable (NSC) region n

Bit number	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
ID																								B					A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	REGION			Region number
B	RW	LOCK			
			Unlocked	0	This register can be updated
			Locked	1	The content of this register can't be changed until the next reset

6.15.9.19 RAMNSC[n].SIZE (n=0..1)

Address offset: 0x544 + (n × 0x8)

Define the size of the non-secure callable (NSC) region n

Bit number	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
ID																								B					A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	SIZE			Size of the non-secure callable (NSC) region n
			Disabled	0	The region n is not defined as a non-secure callable region. Normal security attributes (secure or non-secure) are enforced.
			32	1	The region n is defined as non-secure callable with a 32-byte size
			64	2	The region n is defined as non-secure callable with a 64-byte size
			128	3	The region n is defined as non-secure callable with a 128-byte size
			256	4	The region n is defined as non-secure callable with a 256-byte size
			512	5	The region n is defined as non-secure callable with a 512-byte size
			1024	6	The region n is defined as non-secure callable with a 1024-byte size
			2048	7	The region n is defined as non-secure callable with a 2048-byte size
			4096	8	The region n is defined as non-secure callable with a 4096-byte size
B	RW	LOCK			
			Unlocked	0	This register can be updated
			Locked	1	The content of this register can't be changed until the next reset

6.15.9.20 FLASHREGION[n].PERM (n=0..31)

Address offset: 0x600 + (n × 0x4)

Access permissions for flash region n

Bit number	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
ID																								E				D		C	B	A
Reset 0x00000017	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	EXECUTE			Configure instruction fetch permissions from flash region n
			Enable	1	Allow instruction fetches from flash region n
			Disable	0	Block instruction fetches from flash region n
B	RW	WRITE			Configure write permission for flash region n
			Enable	1	Allow write operation to region n
			Disable	0	Block write operation to region n
C	RW	READ			Configure read permissions for flash region n
			Enable	1	Allow read operation from flash region n
			Disable	0	Block read operation from flash region n
D	RW	SECATTR			Security attribute for flash region n
			Non_Secure	0	Flash region n security attribute is non-secure
			Secure	1	Flash region n security attribute is secure
E	RW	LOCK			
			Unlocked	0	This register can be updated

Bit number	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
ID																								E				D		C	B	A
Reset 0x00000017	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
			Locked	1	The content of this register can't be changed until the next reset

6.15.9.21 RAMREGION[n].PERM (n=0..31)

Address offset: 0x700 + (n × 0x4)

Access permissions for RAM region n

Bit number	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
ID																								E				D		C	B	A
Reset 0x00000017	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	EXECUTE			Configure instruction fetch permissions from RAM region n
			Enable	1	Allow instruction fetches from RAM region n
			Disable	0	Block instruction fetches from RAM region n
B	RW	WRITE			Configure write permission for RAM region n
			Enable	1	Allow write operation to RAM region n
			Disable	0	Block write operation to RAM region n
C	RW	READ			Configure read permissions for RAM region n
			Enable	1	Allow read operation from RAM region n
			Disable	0	Block read operation from RAM region n
D	RW	SECATTR			Security attribute for RAM region n
			Non_Secure	0	RAM region n security attribute is non-secure
			Secure	1	RAM region n security attribute is secure
E	RW	LOCK			
			Unlocked	0	This register can be updated
			Locked	1	The content of this register can't be changed until the next reset

6.15.9.22 PERIPHID[n].PERM (n=0..66)

Address offset: 0x800 + (n × 0x4)

List capabilities and access permissions for the peripheral with ID n

Reset values are unique per peripheral instantation. Please refer to the peripheral instantiation table. Entries not listed in the instantiation table are undefined.

Bit number	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
ID	F																							E			D	C	B	B	A	A
Reset 0x00000012	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**1**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	SECUREMAPPING			Define configuration capabilities for TrustZone Cortex-M secure attribute **Note:** This field is read only
			NonSecure	0	This peripheral is always accessible as a non-secure peripheral
			Secure	1	This peripheral is always accessible as a secure peripheral

Bit number	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
ID	F																							E			D	C	B	B	A	A
Reset 0x00000012	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**1**	**0**

ID	Acce	Field	Value ID	Value	Description
			UserSelectable	2	Non-secure or secure attribute for this peripheral is defined by the PERIPHID[n].PERM register
			Split	3	This peripheral implements the split security mechanism. Non-secure or secure attribute for this peripheral is defined by the PERIPHID[n].PERM register.
B	R	DMA			Indicate if the peripheral has DMA capabilities and if DMA transfer can be assigned to a different security attribute than the peripheral itself
			NoDMA	0	Peripheral has no DMA capability
			NoSeparateAttribute	1	Peripheral has DMA and DMA transfers always have the same security attribute as assigned to the peripheral
			SeparateAttribute	2	Peripheral has DMA and DMA transfers can have a different security attribute than the one assigned to the peripheral
C	RW	SECATTR			Peripheral security mapping **Note:** This bit has effect only if PERIPHID[n].PERM.SECUREMAPPING reads as UserSelectable or Split
			Secure	1	Peripheral is mapped in secure peripheral address space
			NonSecure	0	If SECUREMAPPING == UserSelectable: Peripheral is mapped in non-secure peripheral address space. If SECUREMAPPING == Split: Peripheral is mapped in non-secure and secure peripheral address space.
D	RW	DMASEC			Security attribution for the DMA transfer **Note:** This bit has effect only if PERIPHID[n].PERM.SECATTR is set to secure
			Secure	1	DMA transfers initiated by this peripheral have the secure attribute set
			NonSecure	0	DMA transfers initiated by this peripheral have the non-secure attribute set
E	RW	LOCK			
			Unlocked	0	This register can be updated
			Locked	1	The content of this register can't be changed until the next reset
F	R	PRESENT			Indicate if a peripheral is present with ID n
			NotPresent	0	Peripheral is not present
			IsPresent	1	Peripheral is present

6.16 TIMER — Timer/counter

This peripheral is a general purpose timer designed to keep track of time in user-selective time intervals, it can operate in two modes: timer and counter.

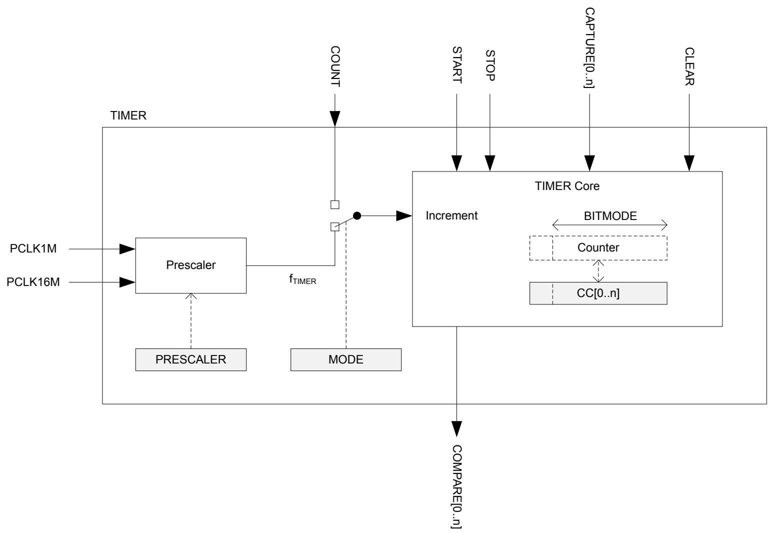

Figure 88: Block schematic for timer/counter

The timer/counter runs on the high-frequency clock source (HFCLK) and includes a four-bit (1/2X) prescaler that can divide the timer input clock from the HFCLK controller. Clock source selection between PCLK16M and PCLK1M is automatic according to TIMER base frequency set by the prescaler. The TIMER base frequency is always given as 16 MHz divided by the prescaler value.

The PPI system allows a TIMER event to trigger a task of any other system peripheral of the device. The PPI system also enables the TIMER task/event features to generate periodic output and PWM signals to any GPIO. The number of input/outputs used at the same time is limited by the number of GPIOTE channels.

The TIMER can operate in two modes, Timer mode and Counter mode. In both modes, the TIMER is started by triggering the START task, and stopped by triggering the STOP task. After the timer is stopped the timer can resume timing/counting by triggering the START task again. When timing/counting is resumed, the timer will continue from the value it had prior to being stopped.

In Timer mode, the TIMER's internal Counter register is incremented by one for every tick of the timer frequency f_{TIMER} as illustrated in Block schematic for timer/counter on page 284. The timer frequency is derived from PCLK16M as shown below, using the values specified in the PRESCALER register:

$$f_{TIMER} = 16\ MHz\ /\ (2^{PRESCALER})$$

When f_{TIMER} <= 1 MHz the TIMER will use PCLK1M instead of PCLK16M for reduced power consumption.

In counter mode, the TIMER's internal Counter register is incremented by one each time the COUNT task is triggered, that is, the timer frequency and the prescaler are not utilized in counter mode. Similarly, the COUNT task has no effect in Timer mode.

The TIMER's maximum value is configured by changing the bit-width of the timer in the BITMODE on page 292 register.

PRESCALER on page 292 and the BITMODE on page 292 must only be updated when the timer is stopped. If these registers are updated while the TIMER is started then this may result in unpredictable behavior.

When the timer is incremented beyond its maximum value the Counter register will overflow and the TIMER will automatically start over from zero.

The Counter register can be cleared, that is, its internal value set to zero explicitly, by triggering the CLEAR task.

The TIMER implements multiple capture/compare registers.

Independent of prescaler setting the accuracy of the TIMER is equivalent to one tick of the timer frequency f_{TIMER} as illustrated in Block schematic for timer/counter on page 284.

6.16.1 Capture

The TIMER implements one capture task for every available capture/compare register.

Every time the CAPTURE[n] task is triggered, the Counter value is copied to the CC[n] register.

6.16.2 Compare

The TIMER implements one COMPARE event for every available capture/compare register.

A COMPARE event is generated when the Counter is incremented and then becomes equal to the value specified in one of the capture compare registers. When the Counter value becomes equal to the value specified in a capture compare register CC[n], the corresponding compare event COMPARE[n] is generated.

BITMODE on page 292 specifies how many bits of the Counter register and the capture/compare register that are used when the comparison is performed. Other bits will be ignored.

The COMPARE event can be configured to operate in one-shot mode by configuring the corresponding ONESHOTEN[n] register. COMPARE[n] event is generated the first time the Counter matches CC[n] after CC[n] has been written.

6.16.3 Task delays

After the TIMER is started, the CLEAR task, COUNT task and the STOP task will guarantee to take effect within one clock cycle of the PCLK16M.

6.16.4 Task priority

If the START task and the STOP task are triggered at the same time, that is, within the same period of PCLK16M, the STOP task will be prioritized.

If one or more of the CAPTURE tasks and the CLEAR task is triggered at the same time, that is, within the same period of PCLK16M, the CLEAR task will be prioritized. This means that the CC register for the relevant CAPTURE task will be set to 0.

6.16.5 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x5000F000 0x4000F000	TIMER	TIMER0 : S TIMER0 : NS	US	NA	Timer 0	
0x50010000 0x40010000	TIMER	TIMER1 : S TIMER1 : NS	US	NA	Timer 1	
0x50011000 0x40011000	TIMER	TIMER2 : S TIMER2 : NS	US	NA	Timer 2	

Table 86: Instances

Register	Offset	Security	Description
TASKS_START	0x000		Start Timer

Register	Offset	Security	Description	
TASKS_STOP	0x004		Stop Timer	
TASKS_COUNT	0x008		Increment Timer (Counter mode only)	
TASKS_CLEAR	0x00C		Clear time	
TASKS_SHUTDOWN	0x010		Shut down timer	Deprecated
TASKS_CAPTURE[0]	0x040		Capture Timer value to CC[0] register	
TASKS_CAPTURE[1]	0x044		Capture Timer value to CC[1] register	
TASKS_CAPTURE[2]	0x048		Capture Timer value to CC[2] register	
TASKS_CAPTURE[3]	0x04C		Capture Timer value to CC[3] register	
TASKS_CAPTURE[4]	0x050		Capture Timer value to CC[4] register	
TASKS_CAPTURE[5]	0x054		Capture Timer value to CC[5] register	
SUBSCRIBE_START	0x080		Subscribe configuration for task START	
SUBSCRIBE_STOP	0x084		Subscribe configuration for task STOP	
SUBSCRIBE_COUNT	0x088		Subscribe configuration for task COUNT	
SUBSCRIBE_CLEAR	0x08C		Subscribe configuration for task CLEAR	
SUBSCRIBE_SHUTDOWN	0x090		Subscribe configuration for task SHUTDOWN	Deprecated
SUBSCRIBE_CAPTURE[0]	0x0C0		Subscribe configuration for task CAPTURE[0]	
SUBSCRIBE_CAPTURE[1]	0x0C4		Subscribe configuration for task CAPTURE[1]	
SUBSCRIBE_CAPTURE[2]	0x0C8		Subscribe configuration for task CAPTURE[2]	
SUBSCRIBE_CAPTURE[3]	0x0CC		Subscribe configuration for task CAPTURE[3]	
SUBSCRIBE_CAPTURE[4]	0x0D0		Subscribe configuration for task CAPTURE[4]	
SUBSCRIBE_CAPTURE[5]	0x0D4		Subscribe configuration for task CAPTURE[5]	
EVENTS_COMPARE[0]	0x140		Compare event on CC[0] match	
EVENTS_COMPARE[1]	0x144		Compare event on CC[1] match	
EVENTS_COMPARE[2]	0x148		Compare event on CC[2] match	
EVENTS_COMPARE[3]	0x14C		Compare event on CC[3] match	
EVENTS_COMPARE[4]	0x150		Compare event on CC[4] match	
EVENTS_COMPARE[5]	0x154		Compare event on CC[5] match	
PUBLISH_COMPARE[0]	0x1C0		Publish configuration for event COMPARE[0]	
PUBLISH_COMPARE[1]	0x1C4		Publish configuration for event COMPARE[1]	
PUBLISH_COMPARE[2]	0x1C8		Publish configuration for event COMPARE[2]	
PUBLISH_COMPARE[3]	0x1CC		Publish configuration for event COMPARE[3]	
PUBLISH_COMPARE[4]	0x1D0		Publish configuration for event COMPARE[4]	
PUBLISH_COMPARE[5]	0x1D4		Publish configuration for event COMPARE[5]	
SHORTS	0x200		Shortcuts between local events and tasks	
INTENSET	0x304		Enable interrupt	
INTENCLR	0x308		Disable interrupt	
MODE	0x504		Timer mode selection	
BITMODE	0x508		Configure the number of bits used by the TIMER	
PRESCALER	0x510		Timer prescaler register	
ONESHOTEN[0]	0x514		Enable one-shot operation for Capture/Compare channel 0	
ONESHOTEN[1]	0x518		Enable one-shot operation for Capture/Compare channel 1	
ONESHOTEN[2]	0x51C		Enable one-shot operation for Capture/Compare channel 2	
ONESHOTEN[3]	0x520		Enable one-shot operation for Capture/Compare channel 3	
ONESHOTEN[4]	0x524		Enable one-shot operation for Capture/Compare channel 4	
ONESHOTEN[5]	0x528		Enable one-shot operation for Capture/Compare channel 5	
CC[0]	0x540		Capture/Compare register 0	
CC[1]	0x544		Capture/Compare register 1	
CC[2]	0x548		Capture/Compare register 2	
CC[3]	0x54C		Capture/Compare register 3	
CC[4]	0x550		Capture/Compare register 4	
CC[5]	0x554		Capture/Compare register 5	

Table 87: Register overview

6.16.5.1 TASKS_START

Address offset: 0x000

Start Timer

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_START			Start Timer
			Trigger	1	Trigger task

6.16.5.2 TASKS_STOP

Address offset: 0x004

Stop Timer

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_STOP			Stop Timer
			Trigger	1	Trigger task

6.16.5.3 TASKS_COUNT

Address offset: 0x008

Increment Timer (Counter mode only)

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_COUNT			Increment Timer (Counter mode only)
			Trigger	1	Trigger task

6.16.5.4 TASKS_CLEAR

Address offset: 0x00C

Clear time

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_CLEAR			Clear time
			Trigger	1	Trigger task

6.16.5.5 TASKS_SHUTDOWN (Deprecated)

Address offset: 0x010

Shut down timer

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_SHUTDOWN			Shut down timer Deprecated
			Trigger	1	Trigger task

6.16.5.6 TASKS_CAPTURE[n] (n=0..5)

Address offset: 0x040 + (n × 0x4)

Capture Timer value to CC[n] register

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_CAPTURE			Capture Timer value to CC[n] register
			Trigger	1	Trigger task

6.16.5.7 SUBSCRIBE_START

Address offset: 0x080

Subscribe configuration for task START

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task START will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.16.5.8 SUBSCRIBE_STOP

Address offset: 0x084

Subscribe configuration for task STOP

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STOP will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.16.5.9 SUBSCRIBE_COUNT

Address offset: 0x088

Subscribe configuration for task COUNT

Bit number				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
ID				B																												A	A	A	A
Reset 0x00000000				**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task COUNT will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.16.5.10 SUBSCRIBE_CLEAR

Address offset: 0x08C

Subscribe configuration for task CLEAR

Bit number				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
ID				B																												A	A	A	A
Reset 0x00000000				**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task CLEAR will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.16.5.11 SUBSCRIBE_SHUTDOWN (Deprecated)

Address offset: 0x090

Subscribe configuration for task SHUTDOWN

Bit number				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
ID				B																												A	A	A	A
Reset 0x00000000				**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task SHUTDOWN will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.16.5.12 SUBSCRIBE_CAPTURE[n] (n=0..5)

Address offset: 0x0C0 + (n × 0x4)

Subscribe configuration for task CAPTURE[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task CAPTURE[n] will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.16.5.13 EVENTS_COMPARE[n] (n=0..5)

Address offset: 0x140 + (n × 0x4)

Compare event on CC[n] match

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_COMPARE			Compare event on CC[n] match
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.16.5.14 PUBLISH_COMPARE[n] (n=0..5)

Address offset: 0x1C0 + (n × 0x4)

Publish configuration for event COMPARE[n]

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event COMPARE[n] will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.16.5.15 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

Bit number	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
ID																			L	K	J	I	H	G			F	E	D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-F	RW	COMPARE[i]_CLEAR (i=0..5)			Shortcut between event COMPARE[i] and task CLEAR
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut
G-L	RW	COMPARE[i]_STOP (i=0..5)			Shortcut between event COMPARE[i] and task STOP
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut

6.16.5.16 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID											F	E	D	C	B	A																
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-F	RW	COMPARE[i] (i=0..5)			Write '1' to enable interrupt for event COMPARE[i]
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.16.5.17 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID											F	E	D	C	B	A																
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A-F	RW	COMPARE[i] (i=0..5)			Write '1' to disable interrupt for event COMPARE[i]
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.16.5.18 MODE

Address offset: 0x504

Timer mode selection

Bit number	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
ID																															A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description	
A	RW	MODE			Timer mode	
			Timer	0	Select Timer mode	
			Counter	1	Select Counter mode	Deprecated
			LowPowerCounter	2	Select Low Power Counter mode	

6.16.5.19 BITMODE

Address offset: 0x508

Configure the number of bits used by the TIMER

Bit number	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
ID																															A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	BITMODE			Timer bit width
			16Bit	0	16 bit timer bit width
			08Bit	1	8 bit timer bit width
			24Bit	2	24 bit timer bit width
			32Bit	3	32 bit timer bit width

6.16.5.20 PRESCALER

Address offset: 0x510

Timer prescaler register

Bit number	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
ID																													A	A	A	A
Reset 0x00000004	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PRESCALER		[0..9]	Prescaler value

6.16.5.21 ONESHOTEN[n] (n=0..5)

Address offset: 0x514 + (n × 0x4)

Enable one-shot operation for Capture/Compare channel n

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ONESHOTEN			Enable one-shot operation Configures the corresponding compare-channel for one-shot operation
			Disable	0	Disable one-shot operation Compare event is generated every time the Counter matches CC[n]
			Enable	1	Enable one-shot operation Compare event is generated the first time the Counter matches CC[n] after CC[n] has been written

6.16.5.22 CC[n] (n=0..5)

Address offset: 0x540 + (n × 0x4)

Capture/Compare register n

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CC			Capture/Compare value Only the number of bits indicated by BITMODE will be used by the TIMER.

6.16.6 Electrical specification

6.17 TWIM — I^2C compatible two-wire interface master with EasyDMA

TWI master with EasyDMA (TWIM) is a two-wire half-duplex master which can communicate with multiple slave devices connected to the same bus

Listed here are the main features for TWIM:

- I^2C compatible
- Supported baud rates: 100, 250, 400 kbps
- Support for clock stretching (non I^2C compliant)
- EasyDMA

The two-wire interface can communicate with a bi-directional wired-AND bus with two lines (SCL, SDA). The protocol makes it possible to interconnect up to 127 individually addressable devices. TWIM is not compatible with CBUS.

The GPIOs used for each two-wire interface line can be chosen from any GPIO on the device and are independently configurable. This enables great flexibility in device pinout and efficient use of board space and signal routing.

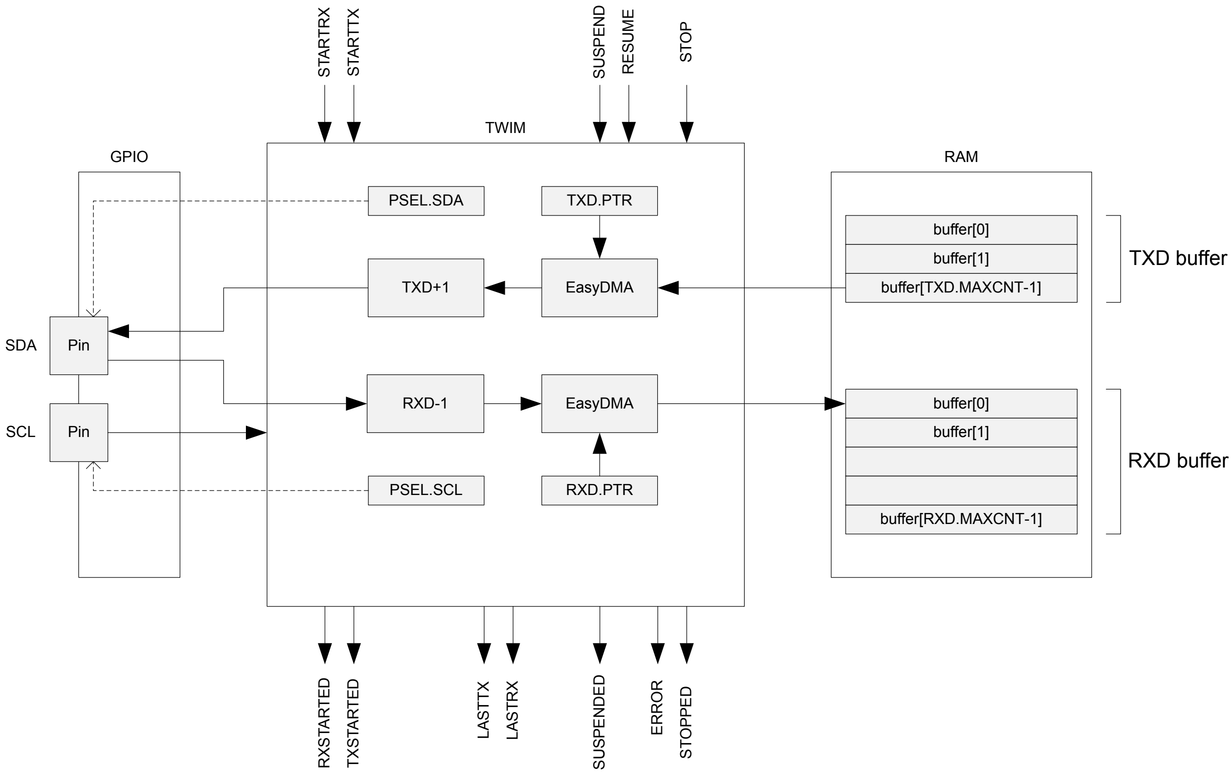

Figure 89: TWI master with EasyDMA

A typical TWI setup consists of one master and one or more slaves. For an example, see A typical TWI setup comprising one master and three slaves on page 294. This TWIM is only able to operate as a single master on the TWI bus. Multi-master bus configuration is not supported.

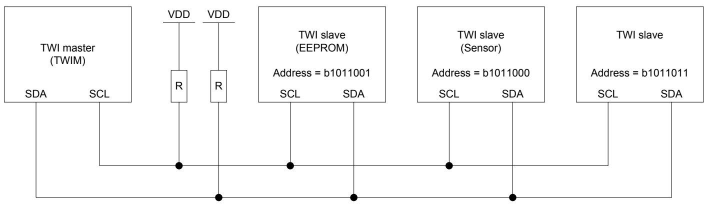

Figure 90: A typical TWI setup comprising one master and three slaves

This TWI master supports clock stretching performed by the slaves. Note that the SCK pulse following a stretched clock cycle may be shorter than specified by the I2C specification.

The TWI master is started by triggering the STARTTX or STARTRX tasks, and stopped by triggering the STOP task. The TWI master will generate a STOPPED event when it has stopped following a STOP task.

After the TWI master is started, the STARTTX or STARTRX tasks should not be triggered again until the TWI master has issued a LASTRX, LASTTX or STOPPED event.

The TWI master can be suspended using the SUSPEND task, this can be used when using the TWI master in a low priority interrupt context. When the TWIM enters suspend state, will automatically issue a SUSPENDED event while performing a continuous clock stretching until it is instructed to resume operation via a RESUME task. The TWI master cannot be stopped while it is suspended, thus the STOP task has to be issued after the TWI master has been resumed.

Note: Any ongoing byte transfer will be allowed to complete before the suspend is enforced. A SUSPEND task has no effect unless the TWI master is actively involved in a transfer.

If a NACK is clocked in from the slave, the TWI master will generate an ERROR event.

6.17.1 Shared resources

The TWI master shares registers and other resources with other peripherals that have the same ID as the TWI master. Therefore, you must disable all peripherals that have the same ID as the TWI master before the TWI master can be configured and used.

Disabling a peripheral that has the same ID as the TWI master will not reset any of the registers that are shared with the TWI master. It is therefore important to configure all relevant registers explicitly to secure that the TWI master operates correctly.

The Instantiation table in Instantiation on page 23 shows which peripherals have the same ID as the TWI.

6.17.2 EasyDMA

The TWIM implements EasyDMA for accessing RAM without CPU involvement.

The TWIM peripheral implements the following EasyDMA channels:

Channel	Type	Register Cluster
TXD	READER	TXD
RXD	WRITER	RXD

Table 88: TWIM EasyDMA Channels

For detailed information regarding the use of EasyDMA, see EasyDMA on page 44.

The .PTR and .MAXCNT registers are double-buffered. They can be updated and prepared for the next RX/TX transmission immediately after having received the RXSTARTED/TXSTARTED event.

The STOPPED event indicates that EasyDMA has finished accessing the buffer in RAM.

6.17.3 Master write sequence

A TWI master write sequence is started by triggering the STARTTX task. After the STARTTX task has been triggered, the TWI master will generate a start condition on the TWI bus, followed by clocking out the address and the READ/WRITE bit set to 0 (WRITE=0, READ=1).

The address must match the address of the slave device that the master wants to write to. The READ/WRITE bit is followed by an ACK/NACK bit (ACK=0 or NACK=1) generated by the slave.

After receiving the ACK bit, the TWI master will clock out the data bytes found in the transmit buffer located in RAM at the address specified in the TXD.PTR register. Each byte clocked out from the master will be followed by an ACK/NACK bit clocked in from the slave.

A typical TWI master write sequence is illustrated in TWI master writing data to a slave on page 296. Occurrence 2 in the figure illustrates clock stretching performed by the TWI master following a SUSPEND task.

A SUSPENDED event indicates that the SUSPEND task has taken effect; this event can be used to synchronize the software.

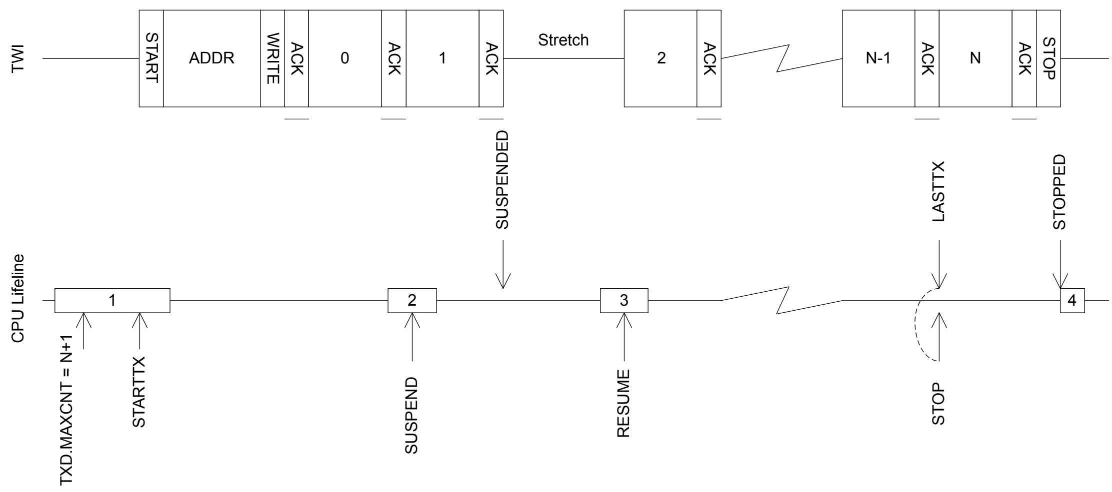

Figure 91: TWI master writing data to a slave

The TWI master will generate a LASTTX event when it starts to transmit the last byte, this is illustrated in TWI master writing data to a slave on page 296

The TWI master is stopped by triggering the STOP task, this task should be triggered during the transmission of the last byte to secure that the TWI master will stop as fast as possible after sending the last byte. The shortcut between LASTTX and STOP can alternatively be used to accomplish this.

> **Note:** The TWI master does not stop by itself when the whole RAM buffer has been sent, or when an error occurs. The STOP task must be issued, through the use of a local or PPI shortcut, or in software as part of the error handler.

6.17.4 Master read sequence

A TWI master read sequence is started by triggering the STARTRX task. After the STARTRX task has been triggered the TWI master will generate a start condition on the TWI bus, followed by clocking out the address and the READ/WRITE bit set to 1 (WRITE = 0, READ = 1). The address must match the address of the slave device that the master wants to read from. The READ/WRITE bit is followed by an ACK/NACK bit (ACK=0 or NACK = 1) generated by the slave.

After having sent the ACK bit the TWI slave will send data to the master using the clock generated by the master.

Data received will be stored in RAM at the address specified in the RXD.PTR register. The TWI master will generate an ACK after all but the last byte received from the slave. The TWI master will generate a NACK after the last byte received to indicate that the read sequence shall stop.

A typical TWI master read sequence is illustrated in The TWI master reading data from a slave on page 297. Occurrence 2 in the figure illustrates clock stretching performed by the TWI master following a SUSPEND task.

A SUSPENDED event indicates that the SUSPEND task has taken effect; this event can be used to synchronize the software.

The TWI master will generate a LASTRX event when it is ready to receive the last byte, this is illustrated in The TWI master reading data from a slave on page 297. If RXD.MAXCNT > 1 the LASTRX event is generated after sending the ACK of the previously received byte. If RXD.MAXCNT = 1 the LASTRX event is generated after receiving the ACK following the address and READ bit.

The TWI master is stopped by triggering the STOP task, this task must be triggered before the NACK bit is supposed to be transmitted. The STOP task can be triggered at any time during the reception of the last byte. It is recommended to use the shortcut between LASTRX and STOP to accomplish this.

Note that the TWI master does not stop by itself when the RAM buffer is full, or when an error occurs. The STOP task must be issued, through the use of a local or PPI shortcut, or in software as part of the error handler.

The TWI master cannot be stopped while it is suspended, so the STOP task has to be issued after the TWI master has been resumed.

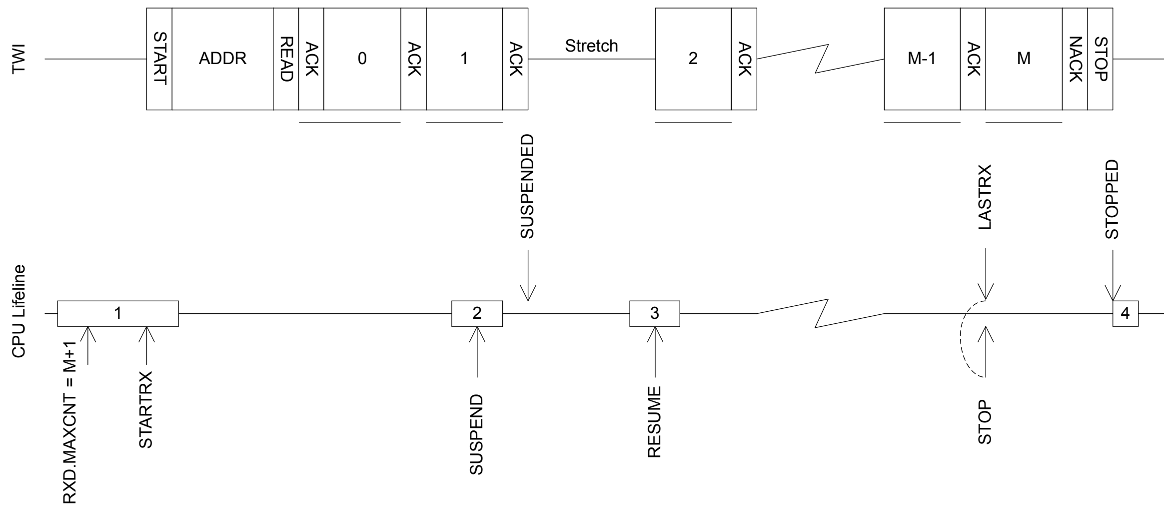

Figure 92: The TWI master reading data from a slave

6.17.5 Master repeated start sequence

A typical repeated start sequence is one in which the TWI master writes two bytes to the slave followed by reading four bytes from the slave. This example uses shortcuts to perform the simplest type of repeated start sequence, i.e. one write followed by one read. The same approach can be used to perform a repeated start sequence where the sequence is read followed by write.

The figure A repeated start sequence, where the TWI master writes two bytes followed by reading 4 bytes from the slave on page 297 illustrates this:

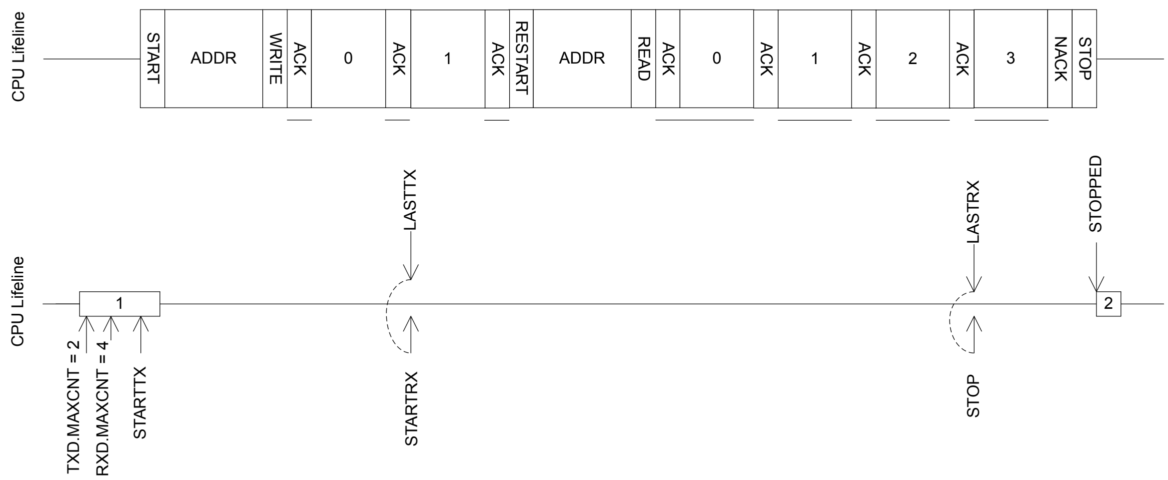

Figure 93: A repeated start sequence, where the TWI master writes two bytes followed by reading 4 bytes from the slave

If a more complex repeated start sequence is needed and the TWI firmware drive is serviced in a low priority interrupt it may be necessary to use the SUSPEND task and SUSPENDED event to guarantee that the correct tasks are generated at the correct time. This is illustrated in A double repeated start sequence using the SUSPEND task to secure recommended operation in low priority interrupts on page 298.

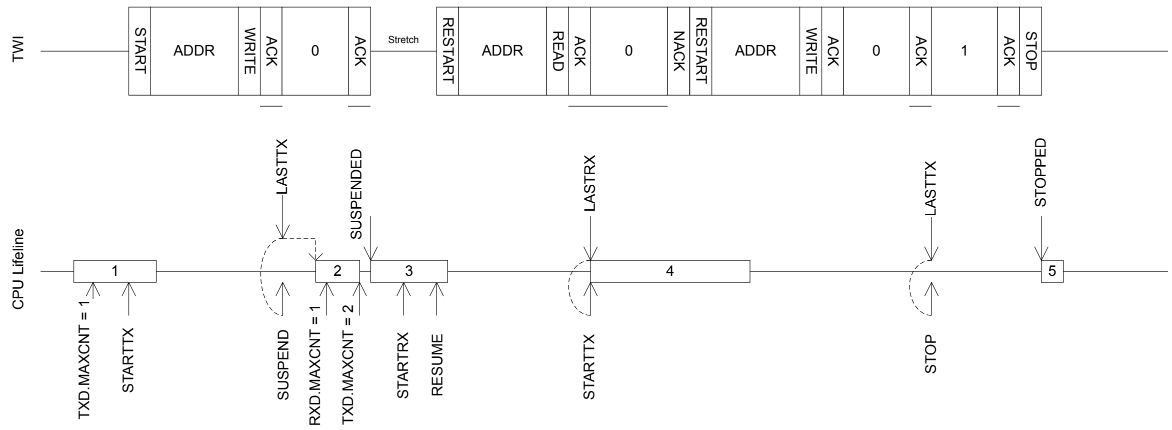

Figure 94: A double repeated start sequence using the SUSPEND task to secure recommended operation in low priority interrupts

6.17.6 Low power

When putting the system in low power and the peripheral is not needed, lowest possible power consumption is achieved by stopping, and then disabling the peripheral.

The STOP task may not be always needed (the peripheral might already be stopped), but if it is sent, software shall wait until the STOPPED event was received as a response before disabling the peripheral through the ENABLE register.

6.17.7 Master mode pin configuration

The SCL and SDA signals associated with the TWI master are mapped to physical pins according to the configuration specified in the PSEL.SCL and PSEL.SDA registers respectively.

The PSEL.SCL and PSEL.SDA registers and their configurations are only used as long as the TWI master is enabled, and retained only as long as the device is in ON mode. When the peripheral is disabled, the pins will behave as regular GPIOs, and use the configuration in their respective OUT bit field and PIN_CNF[n] register. PSEL.SCL, PSEL.SDA must only be configured when the TWI master is disabled.

To secure correct signal levels on the pins used by the TWI master when the system is in OFF mode, and when the TWI master is disabled, these pins must be configured in the GPIO peripheral as described in GPIO configuration before enabling peripheral on page 298.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

TWI master signal	TWI master pin	Direction	Output value	Drive strength
SCL	As specified in PSEL.SCL	Input	Not applicable	S0D1
SDA	As specified in PSEL.SDA	Input	Not applicable	S0D1

Table 89: GPIO configuration before enabling peripheral

6.17.8 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50008000 0x40008000	TWIM	TWIM0 : S TWIM0 : NS	US	SA	Two-wire interface master 0	
0x50009000 0x40009000	TWIM	TWIM1 : S TWIM1 : NS	US	SA	Two-wire interface master 1	
0x5000A000 0x4000A000	TWIM	TWIM2 : S TWIM2 : NS	US	SA	Two-wire interface master 2	
0x5000B000 0x4000B000	TWIM	TWIM3 : S TWIM3 : NS	US	SA	Two-wire interface master 3	

Table 90: Instances

Register	Offset	Security	Description
TASKS_STARTRX	0x000		Start TWI receive sequence
TASKS_STARTTX	0x008		Start TWI transmit sequence
TASKS_STOP	0x014		Stop TWI transaction. Must be issued while the TWI master is not suspended.
TASKS_SUSPEND	0x01C		Suspend TWI transaction
TASKS_RESUME	0x020		Resume TWI transaction
SUBSCRIBE_STARTRX	0x080		Subscribe configuration for task STARTRX
SUBSCRIBE_STARTTX	0x088		Subscribe configuration for task STARTTX
SUBSCRIBE_STOP	0x094		Subscribe configuration for task STOP
SUBSCRIBE_SUSPEND	0x09C		Subscribe configuration for task SUSPEND
SUBSCRIBE_RESUME	0x0A0		Subscribe configuration for task RESUME
EVENTS_STOPPED	0x104		TWI stopped
EVENTS_ERROR	0x124		TWI error
EVENTS_SUSPENDED	0x148		SUSPEND task has been issued, TWI traffic is now suspended.
EVENTS_RXSTARTED	0x14C		Receive sequence started
EVENTS_TXSTARTED	0x150		Transmit sequence started
EVENTS_LASTRX	0x15C		Byte boundary, starting to receive the last byte
EVENTS_LASTTX	0x160		Byte boundary, starting to transmit the last byte
PUBLISH_STOPPED	0x184		Publish configuration for event STOPPED
PUBLISH_ERROR	0x1A4		Publish configuration for event ERROR
PUBLISH_SUSPENDED	0x1C8		Publish configuration for event SUSPENDED
PUBLISH_RXSTARTED	0x1CC		Publish configuration for event RXSTARTED
PUBLISH_TXSTARTED	0x1D0		Publish configuration for event TXSTARTED
PUBLISH_LASTRX	0x1DC		Publish configuration for event LASTRX
PUBLISH_LASTTX	0x1E0		Publish configuration for event LASTTX
SHORTS	0x200		Shortcuts between local events and tasks
INTEN	0x300		Enable or disable interrupt
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
ERRORSRC	0x4C4		Error source
ENABLE	0x500		Enable TWIM
PSEL.SCL	0x508		Pin select for SCL signal
PSEL.SDA	0x50C		Pin select for SDA signal
FREQUENCY	0x524		TWI frequency. Accuracy depends on the HFCLK source selected.
RXD.PTR	0x534		Data pointer
RXD.MAXCNT	0x538		Maximum number of bytes in receive buffer
RXD.AMOUNT	0x53C		Number of bytes transferred in the last transaction
RXD.LIST	0x540		EasyDMA list type
TXD.PTR	0x544		Data pointer

Register	Offset	Security	Description
TXD.MAXCNT	0x548		Maximum number of bytes in transmit buffer
TXD.AMOUNT	0x54C		Number of bytes transferred in the last transaction
TXD.LIST	0x550		EasyDMA list type
ADDRESS	0x588		Address used in the TWI transfer

Table 91: Register overview

6.17.8.1 TASKS_STARTRX

Address offset: 0x000

Start TWI receive sequence

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_STARTRX			Start TWI receive sequence
			Trigger	1	Trigger task

6.17.8.2 TASKS_STARTTX

Address offset: 0x008

Start TWI transmit sequence

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_STARTTX			Start TWI transmit sequence
			Trigger	1	Trigger task

6.17.8.3 TASKS_STOP

Address offset: 0x014

Stop TWI transaction. Must be issued while the TWI master is not suspended.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_STOP			Stop TWI transaction. Must be issued while the TWI master is not suspended.
			Trigger	1	Trigger task

6.17.8.4 TASKS_SUSPEND

Address offset: 0x01C

Suspend TWI transaction

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_SUSPEND			Suspend TWI transaction
			Trigger	1	Trigger task

6.17.8.5 TASKS_RESUME

Address offset: 0x020

Resume TWI transaction

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_RESUME			Resume TWI transaction
			Trigger	1	Trigger task

6.17.8.6 SUBSCRIBE_STARTRX

Address offset: 0x080

Subscribe configuration for task STARTRX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STARTRX will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.17.8.7 SUBSCRIBE_STARTTX

Address offset: 0x088

Subscribe configuration for task STARTTX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STARTTX will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.17.8.8 SUBSCRIBE_STOP

Address offset: 0x094

Subscribe configuration for task STOP

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STOP will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.17.8.9 SUBSCRIBE_SUSPEND

Address offset: 0x09C

Subscribe configuration for task SUSPEND

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task SUSPEND will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.17.8.10 SUBSCRIBE_RESUME

Address offset: 0x0A0

Subscribe configuration for task RESUME

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task RESUME will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.17.8.11 EVENTS_STOPPED

Address offset: 0x104

TWI stopped

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_STOPPED			TWI stopped
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.17.8.12 EVENTS_ERROR

Address offset: 0x124

TWI error

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_ERROR			TWI error
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.17.8.13 EVENTS_SUSPENDED

Address offset: 0x148

SUSPEND task has been issued, TWI traffic is now suspended.

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_SUSPENDED			SUSPEND task has been issued, TWI traffic is now suspended.
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.17.8.14 EVENTS_RXSTARTED

Address offset: 0x14C

Receive sequence started

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_RXSTARTED			Receive sequence started
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.17.8.15 EVENTS_TXSTARTED

Address offset: 0x150

Transmit sequence started

Bit number				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
ID				A
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_TXSTARTED			Transmit sequence started
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.17.8.16 EVENTS_LASTRX

Address offset: 0x15C

Byte boundary, starting to receive the last byte

Bit number				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	
ID				A	
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**	
ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_LASTRX			Byte boundary, starting to receive the last byte
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.17.8.17 EVENTS_LASTTX

Address offset: 0x160

Byte boundary, starting to transmit the last byte

Bit number				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	
ID				A	
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**	
ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_LASTTX			Byte boundary, starting to transmit the last byte
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.17.8.18 PUBLISH_STOPPED

Address offset: 0x184

Publish configuration for event STOPPED

Bit number				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	
ID				B A A A A	
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**	
ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event STOPPED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.17.8.19 PUBLISH_ERROR

Address offset: 0x1A4

Publish configuration for event ERROR

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event ERROR will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.17.8.20 PUBLISH_SUSPENDED

Address offset: 0x1C8

Publish configuration for event SUSPENDED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event SUSPENDED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.17.8.21 PUBLISH_RXSTARTED

Address offset: 0x1CC

Publish configuration for event RXSTARTED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event RXSTARTED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.17.8.22 PUBLISH_TXSTARTED

Address offset: 0x1D0

Publish configuration for event TXSTARTED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event TXSTARTED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.17.8.23 PUBLISH_LASTRX

Address offset: 0x1DC

Publish configuration for event LASTRX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event LASTRX will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.17.8.24 PUBLISH_LASTTX

Address offset: 0x1E0

Publish configuration for event LASTTX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event LASTTX will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.17.8.25 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

Bit number	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
ID																				F	E	D	C	B	A							
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LASTTX_STARTRX			Shortcut between event LASTTX and task STARTRX
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut
B	RW	LASTTX_SUSPEND			Shortcut between event LASTTX and task SUSPEND
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut
C	RW	LASTTX_STOP			Shortcut between event LASTTX and task STOP
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut
D	RW	LASTRX_STARTTX			Shortcut between event LASTRX and task STARTTX
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut
E	RW	LASTRX_SUSPEND			Shortcut between event LASTRX and task SUSPEND
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut

Bit number	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
ID																				F	E	D	C	B	A							
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
F	RW	LASTRX_STOP			Shortcut between event LASTRX and task STOP
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut

6.17.8.26 INTEN

Address offset: 0x300

Enable or disable interrupt

Bit number	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
ID								J	I			H	G	F									D								A	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	STOPPED			Enable or disable interrupt for event STOPPED
			Disabled	0	Disable
			Enabled	1	Enable
D	RW	ERROR			Enable or disable interrupt for event ERROR
			Disabled	0	Disable
			Enabled	1	Enable
F	RW	SUSPENDED			Enable or disable interrupt for event SUSPENDED
			Disabled	0	Disable
			Enabled	1	Enable
G	RW	RXSTARTED			Enable or disable interrupt for event RXSTARTED
			Disabled	0	Disable
			Enabled	1	Enable
H	RW	TXSTARTED			Enable or disable interrupt for event TXSTARTED
			Disabled	0	Disable
			Enabled	1	Enable
I	RW	LASTRX			Enable or disable interrupt for event LASTRX
			Disabled	0	Disable
			Enabled	1	Enable
J	RW	LASTTX			Enable or disable interrupt for event LASTTX
			Disabled	0	Disable
			Enabled	1	Enable

6.17.8.27 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID								J	I			H	G	F									D								A	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	STOPPED			Write '1' to enable interrupt for event STOPPED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
D	RW	ERROR			Write '1' to enable interrupt for event ERROR

Bit number	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
ID								J	I			H	G	F									D								A	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
F	RW	SUSPENDED			Write '1' to enable interrupt for event SUSPENDED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
G	RW	RXSTARTED			Write '1' to enable interrupt for event RXSTARTED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
H	RW	TXSTARTED			Write '1' to enable interrupt for event TXSTARTED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
I	RW	LASTRX			Write '1' to enable interrupt for event LASTRX
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
J	RW	LASTTX			Write '1' to enable interrupt for event LASTTX
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.17.8.28 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID								J	I			H	G	F									D								A	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	STOPPED			Write '1' to disable interrupt for event STOPPED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
D	RW	ERROR			Write '1' to disable interrupt for event ERROR
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
F	RW	SUSPENDED			Write '1' to disable interrupt for event SUSPENDED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
G	RW	RXSTARTED			Write '1' to disable interrupt for event RXSTARTED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

Bit number	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
ID								J	I			H	G	F									D								A	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
H	RW	TXSTARTED			Write '1' to disable interrupt for event TXSTARTED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
I	RW	LASTRX			Write '1' to disable interrupt for event LASTRX
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
J	RW	LASTTX			Write '1' to disable interrupt for event LASTTX
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.17.8.29 ERRORSRC

Address offset: 0x4C4

Error source

Bit number	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
ID																														C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	OVERRUN			Overrun error A new byte was received before previous byte got transferred into RXD buffer. (Previous data is lost)
			NotReceived	0	Error did not occur
			Received	1	Error occurred
B	RW	ANACK			NACK received after sending the address (write '1' to clear)
			NotReceived	0	Error did not occur
			Received	1	Error occurred
C	RW	DNACK			NACK received after sending a data byte (write '1' to clear)
			NotReceived	0	Error did not occur
			Received	1	Error occurred

6.17.8.30 ENABLE

Address offset: 0x500

Enable TWIM

Bit number	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
ID																													A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ENABLE			Enable or disable TWIM
			Disabled	0	Disable TWIM
			Enabled	6	Enable TWIM

6.17.8.31 PSEL.SCL

Address offset: 0x508

Pin select for SCL signal

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.17.8.32 PSEL.SDA

Address offset: 0x50C

Pin select for SDA signal

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.17.8.33 FREQUENCY

Address offset: 0x524

TWI frequency. Accuracy depends on the HFCLK source selected.

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x04000000	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	FREQUENCY			TWI master clock frequency
			K100	0x01980000	100 kbps
			K250	0x04000000	250 kbps
			K400	0x06400000	400 kbps

6.17.8.34 RXD.PTR

Address offset: 0x534

Data pointer

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			Data pointer

> **Note:** See the memory chapter for details about which memories are available for EasyDMA.

6.17.8.35 RXD.MAXCNT

Address offset: 0x538

Maximum number of bytes in receive buffer

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXCNT		[1..0x1FFF]	Maximum number of bytes in receive buffer

6.17.8.36 RXD.AMOUNT

Address offset: 0x53C

Number of bytes transferred in the last transaction

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	AMOUNT		[1..0x1FFF]	Number of bytes transferred in the last transaction. In case of NACK error, includes the NACK'ed byte.

6.17.8.37 RXD.LIST

Address offset: 0x540

EasyDMA list type

Bit number	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
ID																															A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LIST			List type
			Disabled	0	Disable EasyDMA list
			ArrayList	1	Use array list

6.17.8.38 TXD.PTR

Address offset: 0x544

Data pointer

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			Data pointer

Note: See the memory chapter for details about which memories are available for EasyDMA.

6.17.8.39 TXD.MAXCNT

Address offset: 0x548

Maximum number of bytes in transmit buffer

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXCNT		[1..0x1FFF]	Maximum number of bytes in transmit buffer

6.17.8.40 TXD.AMOUNT

Address offset: 0x54C

Number of bytes transferred in the last transaction

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	AMOUNT		[1..0x1FFF]	Number of bytes transferred in the last transaction. In case of NACK error, includes the NACK'ed byte.

6.17.8.41 TXD.LIST

Address offset: 0x550

EasyDMA list type

Bit number	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
ID																															A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LIST			List type
			Disabled	0	Disable EasyDMA list
			ArrayList	1	Use array list

6.17.8.42 ADDRESS

Address offset: 0x588

Address used in the TWI transfer

Bit number	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
ID																										A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ADDRESS			Address used in the TWI transfer

6.17.9 Electrical specification

6.17.9.1 TWIM interface electrical specifications

Symbol	Description	Min.	Typ.	Max.	Units
$f_{TWIM,SCL}$	Bit rates for TWIM[14]	100		400	kbps
$t_{TWIM,START}$	Time from STARTRX/STARTTX task to transmission started		1.615		µs

6.17.9.2 Two Wire Interface Master (TWIM) timing specifications

Symbol	Description	Min.	Typ.	Max.	Units
t_{TWIM,SU_DAT}	Data setup time before positive edge on SCL – all modes	300			ns
t_{TWIM,HD_DAT}	Data hold time after negative edge on SCL – all modes	500			ns
$t_{TWIM,HD\_STA,100kbps}$	TWIM master hold time for START and repeated START condition, 100 kbps	10000			ns
$t_{TWIM,HD\_STA,250kbps}$	TWIM master hold time for START and repeated START condition, 250kbps	4000			ns
$t_{TWIM,HD\_STA,400kbps}$	TWIM master hold time for START and repeated START condition, 400 kbps	2500			ns
$t_{TWIM,SU\_STO,100kbps}$	TWIM master setup time from SCL high to STOP condition, 100 kbps	5000			ns
$t_{TWIM,SU\_STO,250kbps}$	TWIM master setup time from SCL high to STOP condition, 250 kbps	2000			ns
$t_{TWIM,SU\_STO,400kbps}$	TWIM master setup time from SCL high to STOP condition, 400 kbps	1250			ns
$t_{TWIM,BUF,100kbps}$	TWIM master bus free time between STOP and START conditions, 100 kbps	5800			ns
$t_{TWIM,BUF,250kbps}$	TWIM master bus free time between STOP and START conditions, 250 kbps	2700			ns
$t_{TWIM,BUF,400kbps}$	TWIM master bus free time between STOP and START conditions, 400 kbps	2100			ns

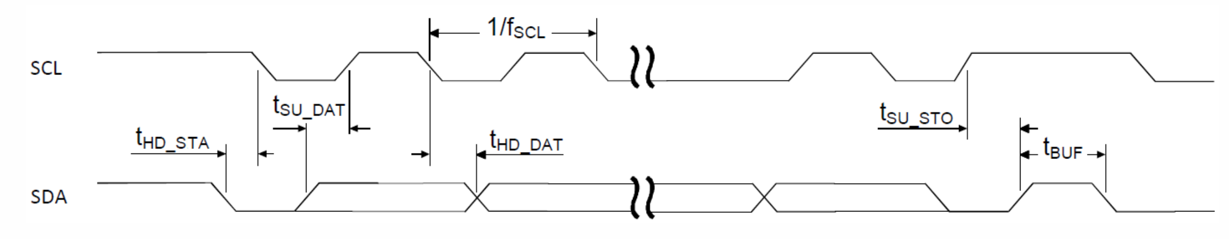

Figure 95: TWIM timing diagram, 1 byte transaction

14 High bit rates or stronger pull-ups may require GPIOs to be set as High Drive, see GPIO chapter for more details.

6.17.10 Pullup resistor

Figure 96: Recommended TWIM pullup value vs. line capacitance

- The I2C specification allows a line capacitance of 400 pF at most.
- The value of internal pullup resistor (R_{PU}) for nRF9160 can be found in GPIO — General purpose input/output on page 95.

6.18 TWIS — I²C compatible two-wire interface slave with EasyDMA

TWI slave with EasyDMA (TWIS) is compatible with I²C operating at 100 kHz and 400 kHz. The TWI transmitter and receiver implement EasyDMA.

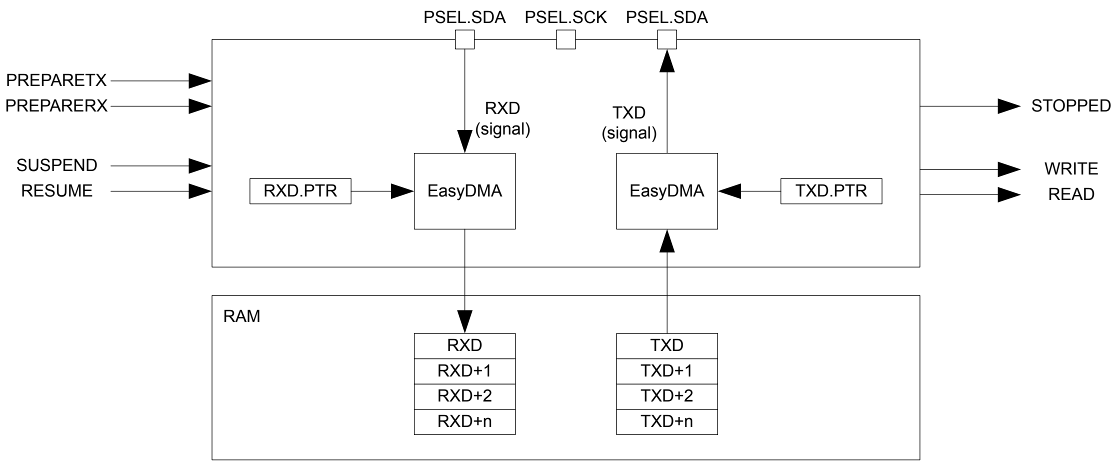

Figure 97: TWI slave with EasyDMA

A typical TWI setup consists of one master and one or more slaves. For an example, see A typical TWI setup comprising one master and three slaves on page 314. TWIS is only able to operate with a single master on the TWI bus.

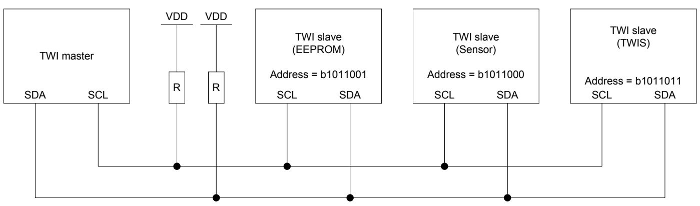

Figure 98: A typical TWI setup comprising one master and three slaves

The TWI slave state machine is illustrated in TWI slave state machine on page 315 and TWI slave state machine symbols on page 315 is explaining the different symbols used in the state machine.

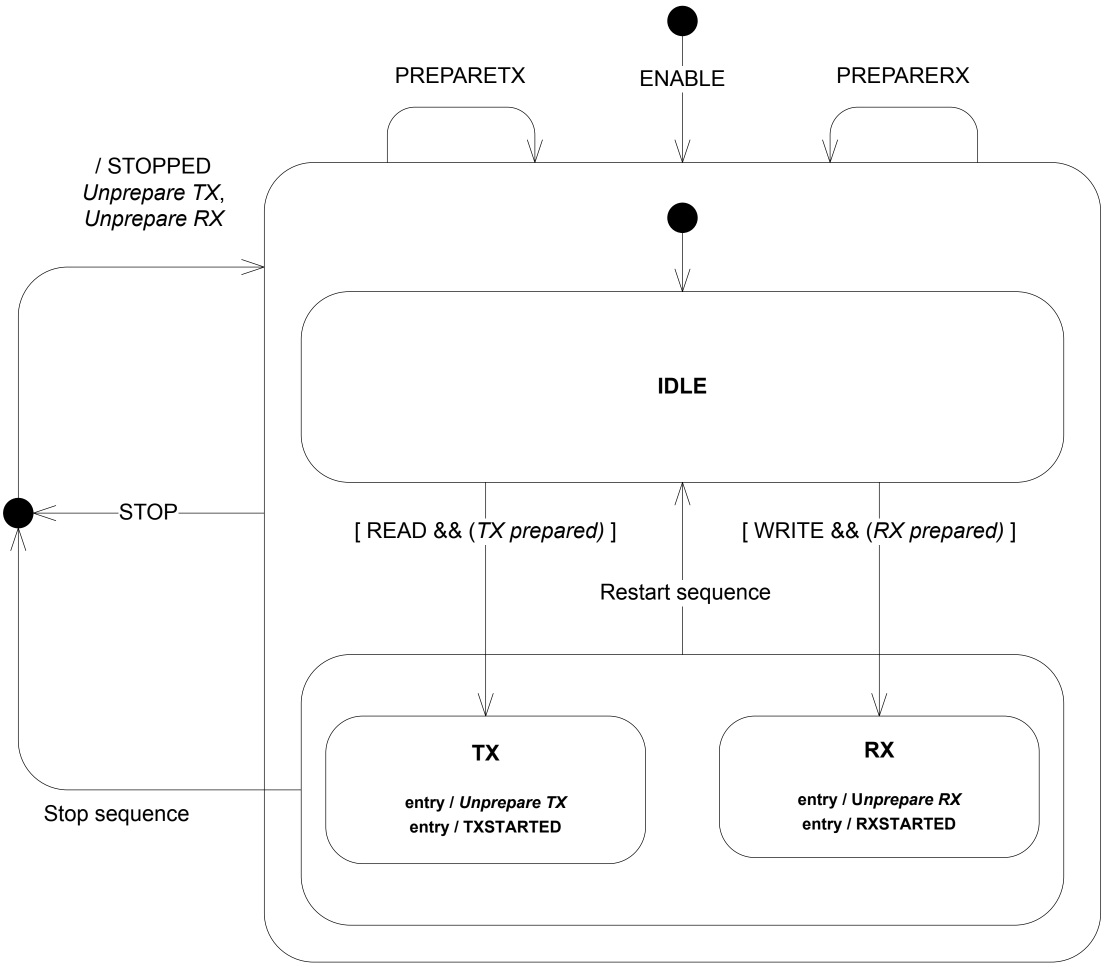

Figure 99: TWI slave state machine

Symbol	Type	Description
ENABLE	Register	The TWI slave has been enabled via the ENABLE register
PREPARETX	Task	The TASKS_PREPARETX task has been triggered
STOP	Task	The TASKS_STOP task has been triggered
PREPARERX	Task	The TASKS_PREPARERX task has been triggered
STOPPED	Event	The EVENTS_STOPPED event was generated
RXSTARTED	Event	The EVENTS_RXSTARTED event was generated
TXSTARTED	Event	The EVENTS_TXSTARTED event was generated
TX prepared	Internal	Internal flag indicating that a TASKS_PREPARETX task has been triggered. This flag is not visible to the user.
RX prepared	Internal	Internal flag indicating that a TASKS_PREPARERX task has been triggered. This flag is not visible to the user.
Unprepare TX	Internal	Clears the internal 'TX prepared' flag until next TASKS_PREPARETX task.
Unprepare RX	Internal	Clears the internal 'RX prepared' flag until next TASKS_PREPARERX task.
Stop condition	TWI protocol	A TWI stop condition was detected
Restart condition	TWI protocol	A TWI restart condition was detected

Table 92: TWI slave state machine symbols

The TWI slave can perform clock stretching, with the premise that the master is able to support it.

The TWI slave operates in a low power mode while waiting for a TWI master to initiate a transfer. As long as the TWI slave is not addressed, it will remain in this low power mode.

To secure correct behaviour of the TWI slave, PSEL.SCL, PSEL.SDA, CONFIG and the ADDRESS[n] registers, must be configured prior to enabling the TWI slave through the ENABLE register. Similarly, changing these settings must be performed while the TWI slave is disabled. Failing to do so may result in unpredictable behaviour.

6.18.1 Shared resources

The TWI slave shares registers and other resources with other peripherals that have the same ID as the TWI slave.

Therefore, you must disable all peripherals that have the same ID as the TWI slave before the TWI slave can be configured and used. Disabling a peripheral that has the same ID as the TWI slave will not reset any of the registers that are shared with the TWI slave. It is therefore important to configure all relevant registers explicitly to secure that the TWI slave operates correctly.

The Instantiation table in Instantiation on page 23 shows which peripherals have the same ID as the TWI slave.

6.18.2 EasyDMA

The TWIS implements EasyDMA for accessing RAM without CPU involvement.

The TWIS peripheral implements the following EasyDMA channels:

Channel	Type	Register Cluster
TXD	READER	TXD
RXD	WRITER	RXD

Table 93: TWIS EasyDMA Channels

For detailed information regarding the use of EasyDMA, see EasyDMA on page 44.

The STOPPED event indicates that EasyDMA has finished accessing the buffer in RAM.

6.18.3 TWI slave responding to a read command

Before the TWI slave can respond to a read command the TWI slave must be configured correctly and enabled via the ENABLE register. When enabled the TWI slave will be in its IDLE state. .

A read command is started when the TWI master generates a start condition on the TWI bus, followed by clocking out the address and the READ/WRITE bit set to 1 (WRITE=0, READ=1). The READ/WRITE bit is followed by an ACK/NACK bit (ACK=0 or NACK=1) response from the TWI slave.

The TWI slave is able to listen for up to two addresses at the same time. Which addresses to listen for is configured in the ADDRESS registers and the CONFIG register.

The TWI slave will only acknowledge (ACK) the read command if the address presented by the master matches one of the addresses the slave is configured to listen for. The TWI slave will generate a READ event when it acknowledges the read command.

The TWI slave is only able to detect a read command from the IDLE state.

The TWI slave will set an internal 'TX prepared' flag when the PREPARETX task is triggered.

When the read command is received the TWI slave will enter the TX state if the internal 'TX prepared' flag is set.

If the internal 'TX prepared' flag is not set when the read command is received, the TWI slave will stretch the master's clock until the PREPARETX task is triggered and the internal 'TX prepared' flag is set.

The TWI slave will generate the TXSTARTED event and clear the 'TX prepared' flag ('unprepare TX') when it enters the TX state. In this state the TWI slave will send the data bytes found in the transmit buffer to the master using the master's clock.

The TWI slave will go back to the IDLE state if the TWI slave receives a restart command when it is in the TX state.

The TWI slave is stopped when it receives the stop condition from the TWI master. A STOPPED event will be generated when the transaction has stopped. The TWI slave will clear the 'TX prepared' flag ('unprepare TX') and go back to the IDLE state when it has stopped.

The transmit buffer is located in RAM at the address specified in the TXD.PTR register. The TWI slave will only be able to send TXD.MAXCNT bytes from the transmit buffer for each transaction. If the TWI master forces the slave to send more than TXD.MAXCNT bytes, the slave will send the byte specified in the ORC register to the master instead. If this happens, an ERROR event will be generated.

The EasyDMA configuration registers, see TXD.PTR etc., are latched when the TXSTARTED event is generated.

The TWI slave can be forced to stop by triggering the STOP task. A STOPPED event will be generated when the TWI slave has stopped. The TWI slave will clear the 'TX prepared' flag and go back to the IDLE state when it has stopped, see also Terminating an ongoing TWI transaction on page 319.

Each byte sent from the slave will be followed by an ACK/NACK bit sent from the master. The TWI master will generate a NACK following the last byte that it wants to receive to tell the slave to release the bus so that the TWI master can generate the stop condition. The TXD.AMOUNT register can be queried after a transaction to see how many bytes were sent.

A typical TWI slave read command response is illustrated in The TWI slave responding to a read command on page 317. Occurrence 2 in the figure illustrates clock stretching performed by the TWI slave following a SUSPEND task.

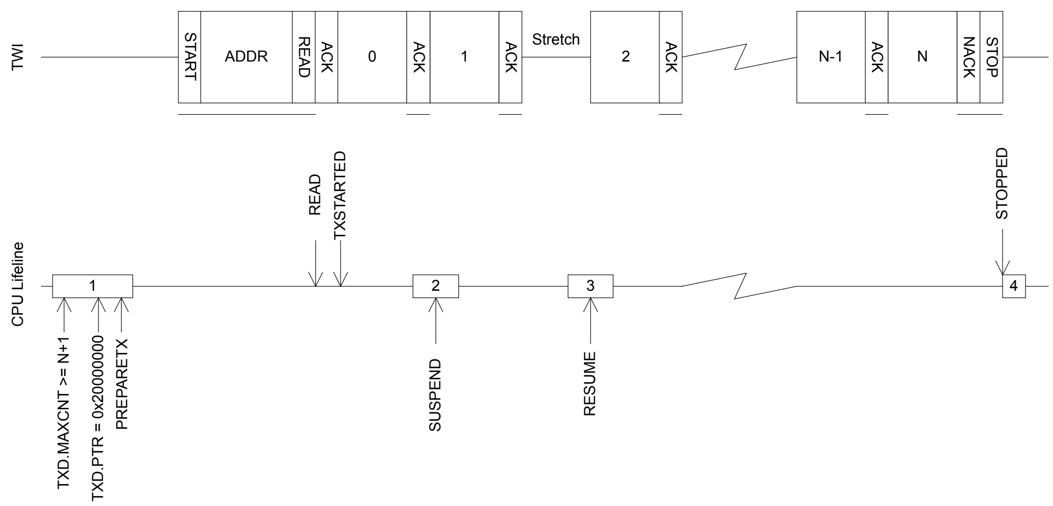

Figure 100: The TWI slave responding to a read command

6.18.4 TWI slave responding to a write command

Before the TWI slave can respond to a write command the TWI slave must be configured correctly and enabled via the ENABLE register. When enabled the TWI slave will be in its IDLE state.

A write command is started when the TWI master generates a start condition on the TWI bus, followed by clocking out the address and the READ/WRITE bit set to 0 (WRITE=0, READ=1). The READ/WRITE bit is followed by an ACK/NACK bit (ACK=0 or NACK=1) response from the slave.

The TWI slave is able to listen for up to two addresses at the same time. Which addresses to listen for is configured in the ADDRESS registers and the CONFIG register.

The TWI slave will only acknowledge (ACK) the write command if the address presented by the master matches one of the addresses the slave is configured to listen for. The TWI slave will generate a WRITE event if it acknowledges the write command.

The TWI slave is only able to detect a write command from the IDLE state.

The TWI slave will set an internal 'RX prepared' flag when the PREPARERX task is triggered.

When the write command is received the TWI slave will enter the RX state if the internal 'RX prepared' flag is set.

If the internal 'RX prepared' flag is not set when the write command is received, the TWI slave will stretch the master's clock until the PREPARERX task is triggered and the internal 'RX prepared' flag is set.

The TWI slave will generate the RXSTARTED event and clear the internal 'RX prepared' flag ('unprepare RX') when it enters the RX state. In this state the TWI slave will be able to receive the bytes sent by the TWI master.

The TWI slave will go back to the IDLE state if the TWI slave receives a restart command when it is in the RX state.

The TWI slave is stopped when it receives the stop condition from the TWI master. A STOPPED event will be generated when the transaction has stopped. The TWI slave will clear the internal 'RX prepared' flag ('unprepare RX') and go back to the IDLE state when it has stopped.

The receive buffer is located in RAM at the address specified in the RXD.PTR register. The TWI slave will only be able to receive as many bytes as specified in the RXD.MAXCNT register. If the TWI master tries to send more bytes to the slave than the slave is able to receive,these bytes will be discarded and the bytes will be NACKed by the slave. If this happens, an ERROR event will be generated.

The EasyDMA configuration registers, see RXD.PTR etc., are latched when the RXSTARTED event is generated.

The TWI slave can be forced to stop by triggering the STOP task. A STOPPED event will be generated when the TWI slave has stopped. The TWI slave will clear the internal 'RX prepared' flag and go back to the IDLE state when it has stopped, see also Terminating an ongoing TWI transaction on page 319.

The TWI slave will generate an ACK after every byte received from the master. The RXD.AMOUNT register can be queried after a transaction to see how many bytes were received.

A typical TWI slave write command response is illustrated in The TWI slave responding to a write command on page 318. Occurrence 2 in the figure illustrates clock stretching performed by the TWI slave following a SUSPEND task.

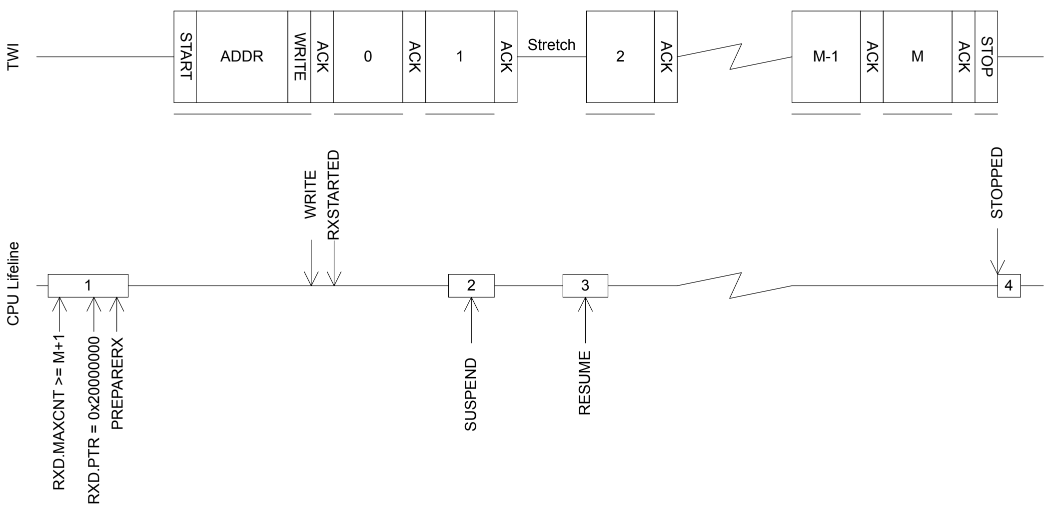

Figure 101: The TWI slave responding to a write command

6.18.5 Master repeated start sequence

An example of a repeated start sequence is one in which the TWI master writes two bytes to the slave followed by reading four bytes from the slave.

This is illustrated in A repeated start sequence, where the TWI master writes two bytes followed by reading four bytes from the slave on page 319.

It is here assumed that the receiver does not know in advance what the master wants to read, and that this information is provided in the first two bytes received in the write part of the repeated start sequence. To guarantee that the CPU is able to process the received data before the TWI slave starts to reply to the read command, the SUSPEND task is triggered via a shortcut from the READ event generated when the read command is received. When the CPU has processed the incoming data and prepared the correct data response, the CPU will resume the transaction by triggering the RESUME task.

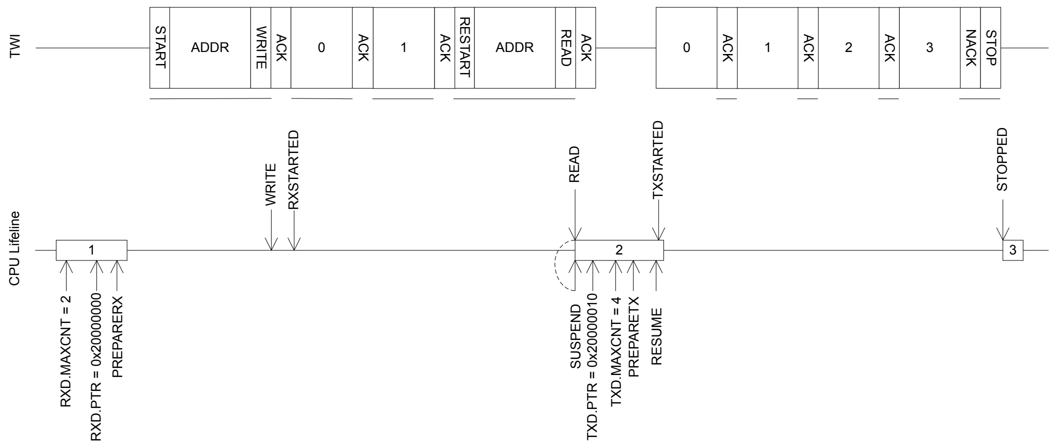

Figure 102: A repeated start sequence, where the TWI master writes two bytes followed by reading four bytes from the slave

6.18.6 Terminating an ongoing TWI transaction

In some situations, e.g. if the external TWI master is not responding correctly, it may be required to terminate an ongoing transaction.

This can be achieved by triggering the STOP task. In this situation a STOPPED event will be generated when the TWI has stopped independent of whether or not a STOP condition has been generated on the TWI bus. The TWI slave will release the bus when it has stopped and go back to its IDLE state.

6.18.7 Low power

When putting the system in low power and the peripheral is not needed, lowest possible power consumption is achieved by stopping, and then disabling the peripheral.

The STOP task may not be always needed (the peripheral might already be stopped), but if it is sent, software shall wait until the STOPPED event was received as a response before disabling the peripheral through the ENABLE register.

6.18.8 Slave mode pin configuration

The SCL and SDA signals associated with the TWI slave are mapped to physical pins according to the configuration specified in the PSEL.SCL and PSEL.SDA registers respectively.

The PSEL.SCL and PSEL.SDA registers and their configurations are only used as long as the TWI slave is enabled, and retained only as long as the device is in ON mode. When the peripheral is disabled, the pins will behave as regular GPIOs, and use the configuration in their respective OUT bit field and PIN_CNF[n] register. PSEL.SCL and PSEL.SDA must only be configured when the TWI slave is disabled.

To secure correct signal levels on the pins used by the TWI slave when the system is in OFF mode, and when the TWI slave is disabled, these pins must be configured in the GPIO peripheral as described in GPIO configuration before enabling peripheral on page 320.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

TWI slave signal	TWI slave pin	Direction	Output value	Drive strength
SCL	As specified in PSEL.SCL	Input	Not applicable	S0D1
SDA	As specified in PSEL.SDA	Input	Not applicable	S0D1

Table 94: GPIO configuration before enabling peripheral

6.18.9 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50008000 0x40008000	TWIS	TWIS0 : S TWIS0 : NS	US	SA	Two-wire interface slave 0	
0x50009000 0x40009000	TWIS	TWIS1 : S TWIS1 : NS	US	SA	Two-wire interface slave 1	
0x5000A000 0x4000A000	TWIS	TWIS2 : S TWIS2 : NS	US	SA	Two-wire interface slave 2	
0x5000B000 0x4000B000	TWIS	TWIS3 : S TWIS3 : NS	US	SA	Two-wire interface slave 3	

Table 95: Instances

Register	Offset	Security	Description
TASKS_STOP	0x014		Stop TWI transaction
TASKS_SUSPEND	0x01C		Suspend TWI transaction
TASKS_RESUME	0x020		Resume TWI transaction
TASKS_PREPARERX	0x030		Prepare the TWI slave to respond to a write command
TASKS_PREPARETX	0x034		Prepare the TWI slave to respond to a read command
SUBSCRIBE_STOP	0x094		Subscribe configuration for task STOP
SUBSCRIBE_SUSPEND	0x09C		Subscribe configuration for task SUSPEND
SUBSCRIBE_RESUME	0x0A0		Subscribe configuration for task RESUME
SUBSCRIBE_PREPARERX	0x0B0		Subscribe configuration for task PREPARERX
SUBSCRIBE_PREPARETX	0x0B4		Subscribe configuration for task PREPARETX
EVENTS_STOPPED	0x104		TWI stopped
EVENTS_ERROR	0x124		TWI error
EVENTS_RXSTARTED	0x14C		Receive sequence started
EVENTS_TXSTARTED	0x150		Transmit sequence started
EVENTS_WRITE	0x164		Write command received
EVENTS_READ	0x168		Read command received
PUBLISH_STOPPED	0x184		Publish configuration for event STOPPED
PUBLISH_ERROR	0x1A4		Publish configuration for event ERROR
PUBLISH_RXSTARTED	0x1CC		Publish configuration for event RXSTARTED
PUBLISH_TXSTARTED	0x1D0		Publish configuration for event TXSTARTED
PUBLISH_WRITE	0x1E4		Publish configuration for event WRITE
PUBLISH_READ	0x1E8		Publish configuration for event READ
SHORTS	0x200		Shortcuts between local events and tasks
INTEN	0x300		Enable or disable interrupt
INTENSET	0x304		Enable interrupt

Register	Offset	Security	Description
INTENCLR	0x308		Disable interrupt
ERRORSRC	0x4D0		Error source
MATCH	0x4D4		Status register indicating which address had a match
ENABLE	0x500		Enable TWIS
PSEL.SCL	0x508		Pin select for SCL signal
PSEL.SDA	0x50C		Pin select for SDA signal
RXD.PTR	0x534		RXD Data pointer
RXD.MAXCNT	0x538		Maximum number of bytes in RXD buffer
RXD.AMOUNT	0x53C		Number of bytes transferred in the last RXD transaction
RXD.LIST	0x540		EasyDMA list type
TXD.PTR	0x544		TXD Data pointer
TXD.MAXCNT	0x548		Maximum number of bytes in TXD buffer
TXD.AMOUNT	0x54C		Number of bytes transferred in the last TXD transaction
TXD.LIST	0x550		EasyDMA list type
ADDRESS[0]	0x588		TWI slave address 0
ADDRESS[1]	0x58C		TWI slave address 1
CONFIG	0x594		Configuration register for the address match mechanism
ORC	0x5C0		Over-read character. Character sent out in case of an over-read of the transmit buffer.

Table 96: Register overview

6.18.9.1 TASKS_STOP

Address offset: 0x014

Stop TWI transaction

Bit number				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
ID				A
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_STOP			Stop TWI transaction
			Trigger	1	Trigger task

6.18.9.2 TASKS_SUSPEND

Address offset: 0x01C

Suspend TWI transaction

Bit number				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
ID				A
Reset 0x00000000				**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_SUSPEND			Suspend TWI transaction
			Trigger	1	Trigger task

6.18.9.3 TASKS_RESUME

Address offset: 0x020

Resume TWI transaction

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_RESUME			Resume TWI transaction
			Trigger	1	Trigger task

6.18.9.4 TASKS_PREPARERX

Address offset: 0x030

Prepare the TWI slave to respond to a write command

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_PREPARERX			Prepare the TWI slave to respond to a write command
			Trigger	1	Trigger task

6.18.9.5 TASKS_PREPARETX

Address offset: 0x034

Prepare the TWI slave to respond to a read command

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_PREPARETX			Prepare the TWI slave to respond to a read command
			Trigger	1	Trigger task

6.18.9.6 SUBSCRIBE_STOP

Address offset: 0x094

Subscribe configuration for task STOP

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STOP will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.18.9.7 SUBSCRIBE_SUSPEND

Address offset: 0x09C

Subscribe configuration for task SUSPEND

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task SUSPEND will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.18.9.8 SUBSCRIBE_RESUME

Address offset: 0x0A0

Subscribe configuration for task RESUME

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task RESUME will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.18.9.9 SUBSCRIBE_PREPARERX

Address offset: 0x0B0

Subscribe configuration for task PREPARERX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task PREPARERX will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.18.9.10 SUBSCRIBE_PREPARETX

Address offset: 0x0B4

Subscribe configuration for task PREPARETX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task PREPARETX will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.18.9.11 EVENTS_STOPPED

Address offset: 0x104

TWI stopped

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_STOPPED			TWI stopped
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.18.9.12 EVENTS_ERROR

Address offset: 0x124

TWI error

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_ERROR			TWI error
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.18.9.13 EVENTS_RXSTARTED

Address offset: 0x14C

Receive sequence started

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_RXSTARTED			Receive sequence started
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.18.9.14 EVENTS_TXSTARTED

Address offset: 0x150

Transmit sequence started

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_TXSTARTED			Transmit sequence started
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.18.9.15 EVENTS_WRITE

Address offset: 0x164

Write command received

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_WRITE			Write command received
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.18.9.16 EVENTS_READ

Address offset: 0x168

Read command received

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_READ			Read command received
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.18.9.17 PUBLISH_STOPPED

Address offset: 0x184

Publish configuration for event STOPPED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event STOPPED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.18.9.18 PUBLISH_ERROR

Address offset: 0x1A4

Publish configuration for event ERROR

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event ERROR will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.18.9.19 PUBLISH_RXSTARTED

Address offset: 0x1CC

Publish configuration for event RXSTARTED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event RXSTARTED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.18.9.20 PUBLISH_TXSTARTED

Address offset: 0x1D0

Publish configuration for event TXSTARTED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event TXSTARTED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.18.9.21 PUBLISH_WRITE

Address offset: 0x1E4

Publish configuration for event WRITE

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event WRITE will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.18.9.22 PUBLISH_READ

Address offset: 0x1E8

Publish configuration for event READ

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event READ will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.18.9.23 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

Bit number	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
ID																		B	A													
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	WRITE_SUSPEND			Shortcut between event WRITE and task SUSPEND
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut
B	RW	READ_SUSPEND			Shortcut between event READ and task SUSPEND
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut

6.18.9.24 INTEN

Address offset: 0x300

Enable or disable interrupt

Bit number	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
ID						H	G					F	E										B								A	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	STOPPED			Enable or disable interrupt for event STOPPED
			Disabled	0	Disable
			Enabled	1	Enable
B	RW	ERROR			Enable or disable interrupt for event ERROR
			Disabled	0	Disable
			Enabled	1	Enable
E	RW	RXSTARTED			Enable or disable interrupt for event RXSTARTED
			Disabled	0	Disable
			Enabled	1	Enable
F	RW	TXSTARTED			Enable or disable interrupt for event TXSTARTED
			Disabled	0	Disable
			Enabled	1	Enable
G	RW	WRITE			Enable or disable interrupt for event WRITE

Bit number	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
ID						H	G					F	E										B								A	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			Disabled	0	Disable
			Enabled	1	Enable
H	RW	READ			Enable or disable interrupt for event READ
			Disabled	0	Disable
			Enabled	1	Enable

6.18.9.25 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID						H	G					F	E										B								A	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	STOPPED			Write '1' to enable interrupt for event STOPPED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	ERROR			Write '1' to enable interrupt for event ERROR
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
E	RW	RXSTARTED			Write '1' to enable interrupt for event RXSTARTED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
F	RW	TXSTARTED			Write '1' to enable interrupt for event TXSTARTED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
G	RW	WRITE			Write '1' to enable interrupt for event WRITE
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
H	RW	READ			Write '1' to enable interrupt for event READ
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.18.9.26 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID						H	G					F	E										B								A	
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	STOPPED			Write '1' to disable interrupt for event STOPPED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	ERROR			Write '1' to disable interrupt for event ERROR
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
E	RW	RXSTARTED			Write '1' to disable interrupt for event RXSTARTED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
F	RW	TXSTARTED			Write '1' to disable interrupt for event TXSTARTED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
G	RW	WRITE			Write '1' to disable interrupt for event WRITE
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
H	RW	READ			Write '1' to disable interrupt for event READ
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.18.9.27 ERRORSRC

Address offset: 0x4D0

Error source

Bit number	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
ID																													C	B		A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	OVERFLOW			RX buffer overflow detected, and prevented
			NotDetected	0	Error did not occur
			Detected	1	Error occurred
B	RW	DNACK			NACK sent after receiving a data byte
			NotReceived	0	Error did not occur
			Received	1	Error occurred
C	RW	OVERREAD			TX buffer over-read detected, and prevented
			NotDetected	0	Error did not occur
			Detected	1	Error occurred

6.18.9.28 MATCH

Address offset: 0x4D4

Status register indicating which address had a match

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	MATCH		[0..1]	Which of the addresses in {ADDRESS} matched the incoming address

6.18.9.29 ENABLE

Address offset: 0x500

Enable TWIS

Bit number	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
ID																													A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ENABLE			Enable or disable TWIS
			Disabled	0	Disable TWIS
			Enabled	9	Enable TWIS

6.18.9.30 PSEL.SCL

Address offset: 0x508

Pin select for SCL signal

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.18.9.31 PSEL.SDA

Address offset: 0x50C

Pin select for SDA signal

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.18.9.32 RXD.PTR

Address offset: 0x534

RXD Data pointer

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			RXD Data pointer

Note: See the memory chapter for details about which memories are available for EasyDMA.

6.18.9.33 RXD.MAXCNT

Address offset: 0x538

Maximum number of bytes in RXD buffer

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXCNT		[1..0x1FFF]	Maximum number of bytes in RXD buffer

6.18.9.34 RXD.AMOUNT

Address offset: 0x53C

Number of bytes transferred in the last RXD transaction

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	AMOUNT		[1..0x1FFF]	Number of bytes transferred in the last RXD transaction

6.18.9.35 RXD.LIST

Address offset: 0x540

EasyDMA list type

Bit number	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
ID																															A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LIST			List type
			Disabled	0	Disable EasyDMA list
			ArrayList	1	Use array list

6.18.9.36 TXD.PTR

Address offset: 0x544

TXD Data pointer

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			TXD Data pointer

Note: See the memory chapter for details about which memories are available for EasyDMA.

6.18.9.37 TXD.MAXCNT

Address offset: 0x548

Maximum number of bytes in TXD buffer

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXCNT		[1..0x1FFF]	Maximum number of bytes in TXD buffer

6.18.9.38 TXD.AMOUNT

Address offset: 0x54C

Number of bytes transferred in the last TXD transaction

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	AMOUNT		[1..0x1FFF]	Number of bytes transferred in the last TXD transaction

6.18.9.39 TXD.LIST

Address offset: 0x550

EasyDMA list type

Bit number	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
ID																															A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	LIST			List type
			Disabled	0	Disable EasyDMA list
			ArrayList	1	Use array list

6.18.9.40 ADDRESS[n] (n=0..1)

Address offset: 0x588 + (n × 0x4)

TWI slave address n

Bit number	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
ID																										A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ADDRESS			TWI slave address

6.18.9.41 CONFIG

Address offset: 0x594

Configuration register for the address match mechanism

Bit number	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
ID																															B	A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A-B	RW	ADDRESS[i] (i=0..1)			Enable or disable address matching on ADDRESS[i]
			Disabled	0	Disabled
			Enabled	1	Enabled

6.18.9.42 ORC

Address offset: 0x5C0

Over-read character. Character sent out in case of an over-read of the transmit buffer.

Bit number	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
ID																									A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ORC			Over-read character. Character sent out in case of an over-read of the transmit buffer.

6.18.10 Electrical specification

6.18.10.1 TWIS slave timing specifications

Symbol	**Description**	**Min.**	**Typ.**	**Max.**	**Units**
$f_{TWIS,SCL}$	Bit rates for TWIS[15]	100		400	kbps
$t_{TWIS,START}$	Time from PREPARERX/PREPARETX task to ready to receive/transmit		1.5		µs
t_{TWIS,SU_DAT}	Data setup time before positive edge on SCL – all modes	300			ns
t_{TWIS,HD_DAT}	Data hold time after negative edge on SCL – all modes	500			ns
$t_{TWIS,HD\_STA,100kbps}$	TWI slave hold time from for START condition (SDA low to SCL low), 100 kbps	5200			ns
$t_{TWIS,HD\_STA,400kbps}$	TWI slave hold time from for START condition (SDA low to SCL low), 400 kbps	1300			ns
$t_{TWIS,SU\_STO,100kbps}$	TWI slave setup time from SCL high to STOP condition, 100 kbps	5200			ns

15 High bit rates or stronger pull-ups may require GPIOs to be set as High Drive, see GPIO chapter for more details.

Symbol	Description	Min.	Typ.	Max.	Units
$t_{TWIS,SU\_STO,400kbps}$	TWI slave setup time from SCL high to STOP condition, 400 kbps	1300			ns
$t_{TWIS,BUF,100kbps}$	TWI slave bus free time between STOP and START conditions, 100 kbps		4700		ns
$t_{TWIS,BUF,400kbps}$	TWI slave bus free time between STOP and START conditions, 400 kbps		1300		ns

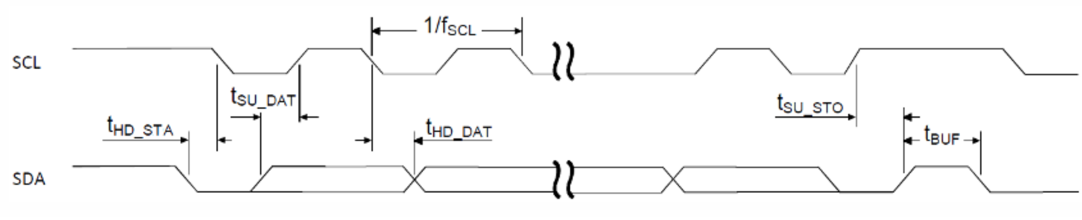

Figure 103: TWIS timing diagram, 1 byte transaction

6.19 UARTE — Universal asynchronous receiver/ transmitter with EasyDMA

The Universal asynchronous receiver/transmitter with EasyDMA (UARTE) offers fast, full-duplex, asynchronous serial communication with built-in flow control (CTS, RTS) support in hardware at a rate up to 1 Mbps, and EasyDMA data transfer from/to RAM.

Listed here are the main features for UARTE:

- Full-duplex operation
- Automatic hardware flow control
- Optional even parity bit checking and generation
- EasyDMA
- Up to 1 Mbps baudrate
- Return to IDLE between transactions supported (when using HW flow control)
- One or two stop bit
- Least significant bit (LSB) first

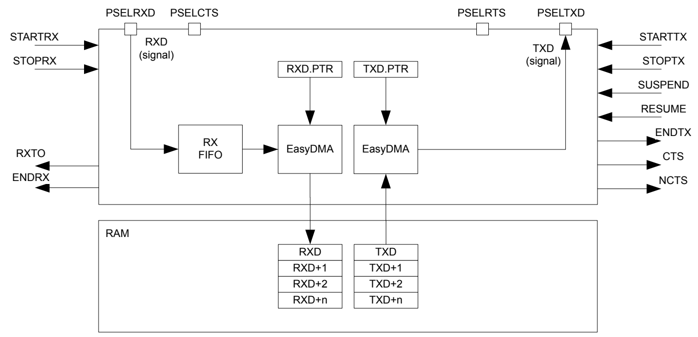

Figure 104: UARTE configuration

The GPIOs used for each UART interface can be chosen from any GPIO on the device and are independently configurable. This enables great flexibility in device pinout and efficient use of board space and signal routing.

6.19.1 EasyDMA

The UARTE implements EasyDMA for reading and writing to and from the RAM.

If the TXD.PTR and the RXD.PTR are not pointing to the Data RAM region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 20 for more information about the different memory regions.

The .PTR and .MAXCNT registers are double-buffered. They can be updated and prepared for the next RX/TX transmission immediately after having received the RXSTARTED/TXSTARTED event.

The ENDRX/ENDTX event indicates that EasyDMA has finished accessing respectively the RX/TX buffer in RAM.

6.19.2 Transmission

The first step of a DMA transmission is storing bytes in the transmit buffer and configuring EasyDMA. This is achieved by writing the initial address pointer to TXD.PTR, and the number of bytes in the RAM buffer to TXD.MAXCNT. The UARTE transmission is started by triggering the STARTTX task.

After each byte has been sent over the TXD line, a TXDRDY event will be generated.

When all bytes in the TXD buffer, as specified in the TXD.MAXCNT register, have been transmitted, the UARTE transmission will end automatically and an ENDTX event will be generated.

A UARTE transmission sequence is stopped by triggering the STOPTX task, a TXSTOPPED event will be generated when the UARTE transmitter has stopped.

If the ENDTX event has not already been generated when the UARTE transmitter has come to a stop, the UARTE will generate the ENDTX event explicitly even though all bytes in the TXD buffer, as specified in the TXD.MAXCNT register, have not been transmitted.

If flow control is enabled through the HWFC field in the CONFIG register, a transmission will be automatically suspended when CTS is deactivated and resumed when CTS is activated again, as illustrated in UARTE transmission on page 335. A byte that is in transmission when CTS is deactivated will be fully transmitted before the transmission is suspended.

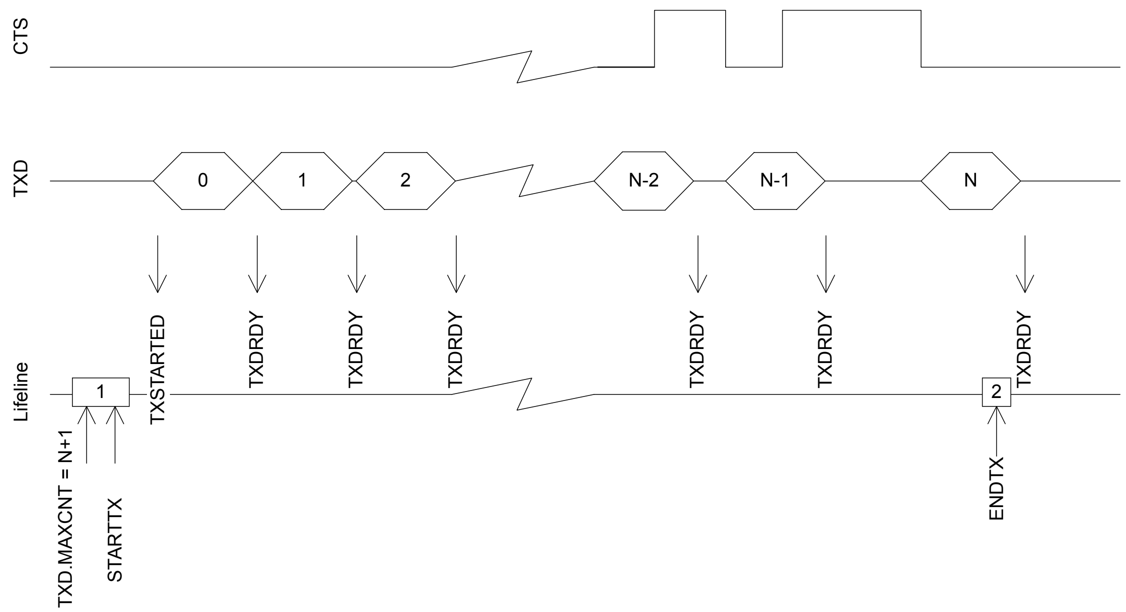

Figure 105: UARTE transmission

The UARTE transmitter will be in its lowest activity level, and consume the least amount of energy, when it is stopped, i.e. before it is started via STARTTX or after it has been stopped via STOPTX and the TXSTOPPED event has been generated. See POWER — Power control on page 59 for more information about power modes.

6.19.3 Reception

The UARTE receiver is started by triggering the STARTRX task. The UARTE receiver is using EasyDMA to store incoming data in an RX buffer in RAM.

The RX buffer is located at the address specified in the RXD.PTR register. The RXD.PTR register is double-buffered and it can be updated and prepared for the next STARTRX task immediately after the RXSTARTED event is generated. The size of the RX buffer is specified in the RXD.MAXCNT register and the UARTE will generate an ENDRX event when it has filled up the RX buffer, see UARTE reception on page 336.

For each byte received over the RXD line, an RXDRDY event will be generated. This event is likely to occur before the corresponding data has been transferred to Data RAM.

The RXD.AMOUNT register can be queried following an ENDRX event to see how many new bytes have been transferred to the RX buffer in RAM since the previous ENDRX event.

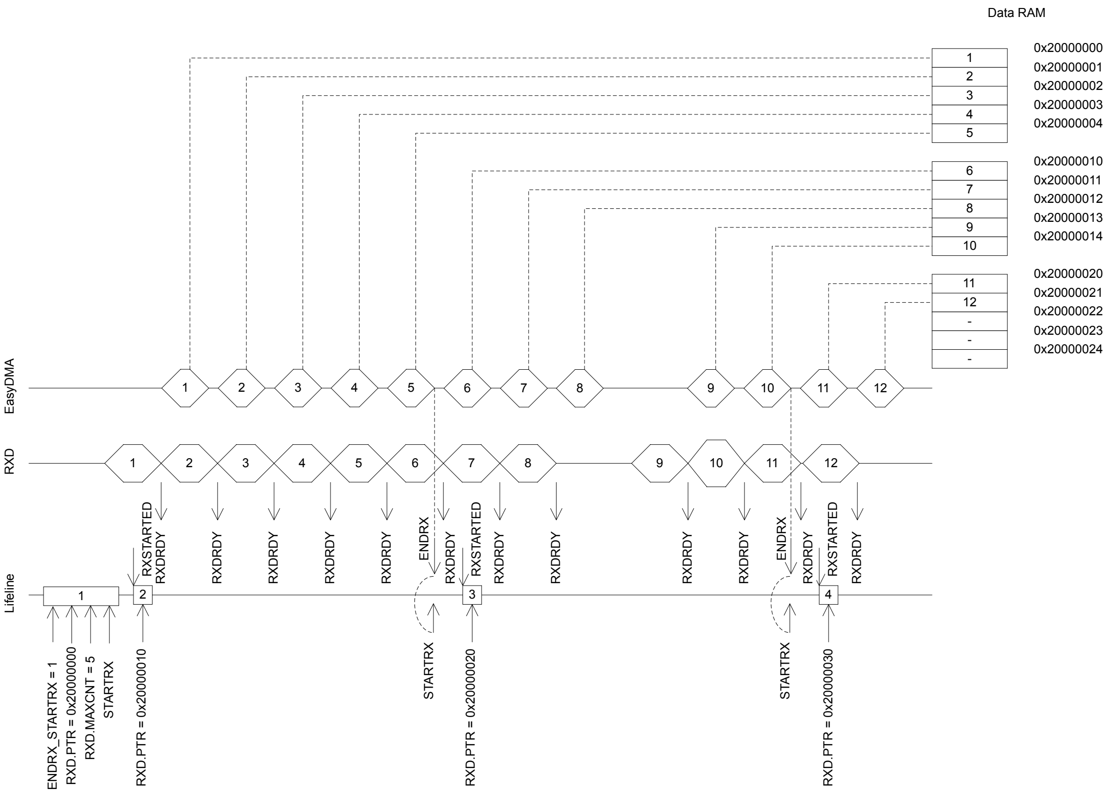

Figure 106: UARTE reception

The UARTE receiver is stopped by triggering the STOPRX task. An RXTO event is generated when the UARTE has stopped. The UARTE will make sure that an impending ENDRX event will be generated before the RXTO event is generated. This means that the UARTE will guarantee that no ENDRX event will be generated after RXTO, unless the UARTE is restarted or a FLUSHRX command is issued after the RXTO event is generated.

> **Important:** If the ENDRX event has not already been generated when the UARTE receiver has come to a stop, which implies that all pending content in the RX FIFO has been moved to the RX buffer, the UARTE will generate the ENDRX event explicitly even though the RX buffer is not full. In this scenario the ENDRX event will be generated before the RXTO event is generated.

To be able to know how many bytes have actually been received into the RX buffer, the CPU can read the RXD.AMOUNT register following the ENDRX event or the RXTO event.

The UARTE is able to receive up to four bytes after the STOPRX task has been triggered as long as these are sent in succession immediately after the RTS signal is deactivated. This is possible because after the RTS is deactivated the UARTE is able to receive bytes for an extended period equal to the time it takes to send 4 bytes on the configured baud rate.

After the RXTO event is generated the internal RX FIFO may still contain data, and to move this data to RAM the FLUSHRX task must be triggered. To make sure that this data does not overwrite data in the RX buffer, the RX buffer should be emptied or the RXD.PTR should be updated before the FLUSHRX task is triggered. To make sure that all data in the RX FIFO is moved to the RX buffer, the RXD.MAXCNT register must be set to RXD.MAXCNT > 4, see UARTE reception with forced stop via STOPRX on page 337. The UARTE will generate the ENDRX event after completing the FLUSHRX task even if the RX FIFO was empty or if the RX buffer does not get filled up. To be able to know how many bytes have actually been received into the RX buffer in this case, the CPU can read the RXD.AMOUNT register following the ENDRX event.

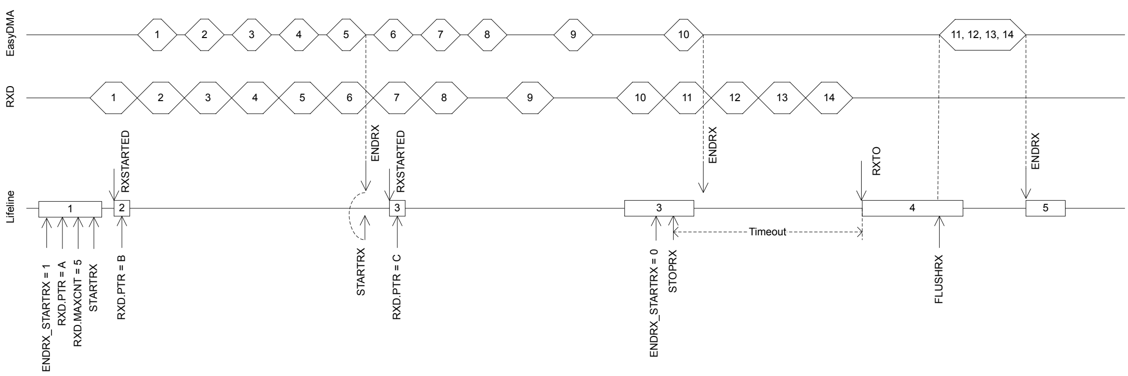

Figure 107: UARTE reception with forced stop via STOPRX

If HW flow control is enabled through the HWFC field in the CONFIG register, the RTS signal will be deactivated when the receiver is stopped via the STOPRX task or when the UARTE is only able to receive four more bytes in its internal RX FIFO.

With flow control disabled, the UARTE will function in the same way as when the flow control is enabled except that the RTS line will not be used. This means that no signal will be generated when the UARTE has reached the point where it is only able to receive four more bytes in its internal RX FIFO. Data received when the internal RX FIFO is filled up, will be lost.

The UARTE receiver will be in its lowest activity level, and consume the least amount of energy, when it is stopped, i.e. before it is started via STARTRX or after it has been stopped via STOPRX and the RXTO event has been generated. See POWER — Power control on page 59 for more information about power modes.

6.19.4 Error conditions

An ERROR event, in the form of a framing error, will be generated if a valid stop bit is not detected in a frame. Another ERROR event, in the form of a break condition, will be generated if the RXD line is held active low for longer than the length of a data frame. Effectively, a framing error is always generated before a break condition occurs.

An ERROR event will not stop reception. If the error was a parity error, the received byte will still be transferred into Data RAM, and so will following incoming bytes. If there was a framing error (wrong stop bit), that specific byte will NOT be stored into Data RAM, but following incoming bytes will.

6.19.5 Using the UARTE without flow control

If flow control is not enabled, the interface will behave as if the CTS and RTS lines are kept active all the time.

6.19.6 Parity and stop bit configuration

When parity is enabled through the PARITY field in the CONFIG register, the parity will be generated automatically from the even parity of TXD and RXD for transmission and reception respectively.

The amount of stop bits can be configured through the STOP field in the CONFIG register.

6.19.7 Low power

When putting the system in low power and the peripheral is not needed, lowest possible power consumption is achieved by stopping, and then disabling the peripheral.

The STOPTX and STOPRX tasks may not be always needed (the peripheral might already be stopped), but if STOPTX and/or STOPRX is sent, software shall wait until the TXSTOPPED and/or RXTO event is received in response, before disabling the peripheral through the ENABLE register.

6.19.8 Pin configuration

The different signals RXD, CTS (Clear To Send, active low), RTS (Request To Send, active low), and TXD associated with the UARTE are mapped to physical pins according to the configuration specified in the PSEL.RXD, PSEL.CTS, PSEL.RTS, and PSEL.TXD registers respectively.

The PSEL.RXD, PSEL.CTS, PSEL.RTS, and PSEL.TXD registers and their configurations are only used as long as the UARTE is enabled, and retained only for the duration the device is in ON mode. PSEL.RXD, PSEL.RTS, PSEL.RTS and PSEL.TXD must only be configured when the UARTE is disabled.

To secure correct signal levels on the pins by the UARTE when the system is in OFF mode, the pins must be configured in the GPIO peripheral as described in GPIO configuration before enabling peripheral on page 338.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

UARTE signal	UARTE pin	Direction	Output value
RXD	As specified in PSEL.RXD	Input	Not applicable
CTS	As specified in PSEL.CTS	Input	Not applicable
RTS	As specified in PSEL.RTS	Output	1
TXD	As specified in PSEL.TXD	Output	1

Table 97: GPIO configuration before enabling peripheral

6.19.9 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50008000 0x40008000	UARTE	UARTE0 : S UARTE0 : NS	US	SA	Universal asynchronous receiver/transmitter with EasyDMA 0	
0x50009000 0x40009000	UARTE	UARTE1 : S UARTE1 : NS	US	SA	Universal asynchronous receiver/transmitter with EasyDMA 1	
0x5000A000 0x4000A000	UARTE	UARTE2 : S UARTE2 : NS	US	SA	Universal asynchronous receiver/transmitter with EasyDMA 2	
0x5000B000 0x4000B000	UARTE	UARTE3 : S UARTE3 : NS	US	SA	Universal asynchronous receiver/transmitter with EasyDMA 3	

Table 98: Instances

Register	Offset	Security	Description
TASKS_STARTRX	0x000		Start UART receiver
TASKS_STOPRX	0x004		Stop UART receiver
TASKS_STARTTX	0x008		Start UART transmitter
TASKS_STOPTX	0x00C		Stop UART transmitter
TASKS_FLUSHRX	0x02C		Flush RX FIFO into RX buffer
SUBSCRIBE_STARTRX	0x080		Subscribe configuration for task STARTRX
SUBSCRIBE_STOPRX	0x084		Subscribe configuration for task STOPRX
SUBSCRIBE_STARTTX	0x088		Subscribe configuration for task STARTTX
SUBSCRIBE_STOPTX	0x08C		Subscribe configuration for task STOPTX
SUBSCRIBE_FLUSHRX	0x0AC		Subscribe configuration for task FLUSHRX
EVENTS_CTS	0x100		CTS is activated (set low). Clear To Send.
EVENTS_NCTS	0x104		CTS is deactivated (set high). Not Clear To Send.
EVENTS_RXDRDY	0x108		Data received in RXD (but potentially not yet transferred to Data RAM)
EVENTS_ENDRX	0x110		Receive buffer is filled up
EVENTS_TXDRDY	0x11C		Data sent from TXD
EVENTS_ENDTX	0x120		Last TX byte transmitted
EVENTS_ERROR	0x124		Error detected
EVENTS_RXTO	0x144		Receiver timeout
EVENTS_RXSTARTED	0x14C		UART receiver has started
EVENTS_TXSTARTED	0x150		UART transmitter has started
EVENTS_TXSTOPPED	0x158		Transmitter stopped
PUBLISH_CTS	0x180		Publish configuration for event CTS
PUBLISH_NCTS	0x184		Publish configuration for event NCTS
PUBLISH_RXDRDY	0x188		Publish configuration for event RXDRDY
PUBLISH_ENDRX	0x190		Publish configuration for event ENDRX
PUBLISH_TXDRDY	0x19C		Publish configuration for event TXDRDY
PUBLISH_ENDTX	0x1A0		Publish configuration for event ENDTX
PUBLISH_ERROR	0x1A4		Publish configuration for event ERROR
PUBLISH_RXTO	0x1C4		Publish configuration for event RXTO
PUBLISH_RXSTARTED	0x1CC		Publish configuration for event RXSTARTED
PUBLISH_TXSTARTED	0x1D0		Publish configuration for event TXSTARTED
PUBLISH_TXSTOPPED	0x1D8		Publish configuration for event TXSTOPPED
SHORTS	0x200		Shortcuts between local events and tasks
INTEN	0x300		Enable or disable interrupt

Register	Offset	Security	Description
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
ERRORSRC	0x480		Error source Note : this register is read / write one to clear.
ENABLE	0x500		Enable UART
PSEL.RTS	0x508		Pin select for RTS signal
PSEL.TXD	0x50C		Pin select for TXD signal
PSEL.CTS	0x510		Pin select for CTS signal
PSEL.RXD	0x514		Pin select for RXD signal
BAUDRATE	0x524		Baud rate. Accuracy depends on the HFCLK source selected.
RXD.PTR	0x534		Data pointer
RXD.MAXCNT	0x538		Maximum number of bytes in receive buffer
RXD.AMOUNT	0x53C		Number of bytes transferred in the last transaction
TXD.PTR	0x544		Data pointer
TXD.MAXCNT	0x548		Maximum number of bytes in transmit buffer
TXD.AMOUNT	0x54C		Number of bytes transferred in the last transaction
CONFIG	0x56C		Configuration of parity and hardware flow control

Table 99: Register overview

6.19.9.1 TASKS_STARTRX

Address offset: 0x000

Start UART receiver

Bit number			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
ID			A
Reset 0x00000000			**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
ID	Acce Field	Value ID	Value	Description
A	W TASKS_STARTRX			Start UART receiver
		Trigger	1	Trigger task

6.19.9.2 TASKS_STOPRX

Address offset: 0x004

Stop UART receiver

Bit number			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
ID			A
Reset 0x00000000			**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
ID	Acce Field	Value ID	Value	Description
A	W TASKS_STOPRX			Stop UART receiver
		Trigger	1	Trigger task

6.19.9.3 TASKS_STARTTX

Address offset: 0x008

Start UART transmitter

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_STARTTX			Start UART transmitter
			Trigger	1	Trigger task

6.19.9.4 TASKS_STOPTX

Address offset: 0x00C

Stop UART transmitter

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_STOPTX			Stop UART transmitter
			Trigger	1	Trigger task

6.19.9.5 TASKS_FLUSHRX

Address offset: 0x02C

Flush RX FIFO into RX buffer

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_FLUSHRX			Flush RX FIFO into RX buffer
			Trigger	1	Trigger task

6.19.9.6 SUBSCRIBE_STARTRX

Address offset: 0x080

Subscribe configuration for task STARTRX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STARTRX will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.19.9.7 SUBSCRIBE_STOPRX

Address offset: 0x084

Subscribe configuration for task STOPRX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STOPRX will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.19.9.8 SUBSCRIBE_STARTTX

Address offset: 0x088

Subscribe configuration for task STARTTX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STARTTX will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.19.9.9 SUBSCRIBE_STOPTX

Address offset: 0x08C

Subscribe configuration for task STOPTX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task STOPTX will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.19.9.10 SUBSCRIBE_FLUSHRX

Address offset: 0x0AC

Subscribe configuration for task FLUSHRX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task FLUSHRX will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.19.9.11 EVENTS_CTS

Address offset: 0x100

CTS is activated (set low). Clear To Send.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_CTS			CTS is activated (set low). Clear To Send.
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.19.9.12 EVENTS_NCTS

Address offset: 0x104

CTS is deactivated (set high). Not Clear To Send.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_NCTS			CTS is deactivated (set high). Not Clear To Send.
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.19.9.13 EVENTS_RXDRDY

Address offset: 0x108

Data received in RXD (but potentially not yet transferred to Data RAM)

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_RXDRDY			Data received in RXD (but potentially not yet transferred to Data RAM)
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.19.9.14 EVENTS_ENDRX

Address offset: 0x110

Receive buffer is filled up

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_ENDRX			Receive buffer is filled up
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.19.9.15 EVENTS_TXDRDY

Address offset: 0x11C

Data sent from TXD

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_TXDRDY			Data sent from TXD
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.19.9.16 EVENTS_ENDTX

Address offset: 0x120

Last TX byte transmitted

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_ENDTX			Last TX byte transmitted
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.19.9.17 EVENTS_ERROR

Address offset: 0x124

Error detected

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_ERROR			Error detected
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.19.9.18 EVENTS_RXTO

Address offset: 0x144

Receiver timeout

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_RXTO			Receiver timeout
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.19.9.19 EVENTS_RXSTARTED

Address offset: 0x14C

UART receiver has started

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_RXSTARTED			UART receiver has started
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.19.9.20 EVENTS_TXSTARTED

Address offset: 0x150

UART transmitter has started

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_TXSTARTED			UART transmitter has started
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.19.9.21 EVENTS_TXSTOPPED

Address offset: 0x158

Transmitter stopped

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_TXSTOPPED			Transmitter stopped
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.19.9.22 PUBLISH_CTS

Address offset: 0x180

Publish configuration for event CTS

Bit number	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
ID	B A A A A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event CTS will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.19.9.23 PUBLISH_NCTS

Address offset: 0x184

Publish configuration for event NCTS

Bit number	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
ID	B A A A A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event NCTS will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.19.9.24 PUBLISH_RXDRDY

Address offset: 0x188

Publish configuration for event RXDRDY

Bit number	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
ID	B A A A A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event RXDRDY will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.19.9.25 PUBLISH_ENDRX

Address offset: 0x190

Publish configuration for event ENDRX

Bit number	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
ID	B A A A A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event ENDRX will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.19.9.26 PUBLISH_TXDRDY

Address offset: 0x19C

Publish configuration for event TXDRDY

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event TXDRDY will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.19.9.27 PUBLISH_ENDTX

Address offset: 0x1A0

Publish configuration for event ENDTX

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event ENDTX will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.19.9.28 PUBLISH_ERROR

Address offset: 0x1A4

Publish configuration for event ERROR

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event ERROR will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.19.9.29 PUBLISH_RXTO

Address offset: 0x1C4

Publish configuration for event RXTO

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event RXTO will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.19.9.30 PUBLISH_RXSTARTED

Address offset: 0x1CC

Publish configuration for event RXSTARTED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event RXSTARTED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.19.9.31 PUBLISH_TXSTARTED

Address offset: 0x1D0

Publish configuration for event TXSTARTED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event TXSTARTED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.19.9.32 PUBLISH_TXSTOPPED

Address offset: 0x1D8

Publish configuration for event TXSTOPPED

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event TXSTOPPED will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.19.9.33 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

Bit number	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
ID																										D	C					
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
C	RW	ENDRX_STARTRX			Shortcut between event ENDRX and task STARTRX
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut
D	RW	ENDRX_STOPRX			Shortcut between event ENDRX and task STOPRX
			Disabled	0	Disable shortcut
			Enabled	1	Enable shortcut

6.19.9.34 INTEN

Address offset: 0x300

Enable or disable interrupt

Bit number	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
ID										L		J	I		H								G	F	E			D		C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CTS			Enable or disable interrupt for event CTS
			Disabled	0	Disable
			Enabled	1	Enable
B	RW	NCTS			Enable or disable interrupt for event NCTS
			Disabled	0	Disable
			Enabled	1	Enable
C	RW	RXDRDY			Enable or disable interrupt for event RXDRDY
			Disabled	0	Disable
			Enabled	1	Enable
D	RW	ENDRX			Enable or disable interrupt for event ENDRX
			Disabled	0	Disable
			Enabled	1	Enable
E	RW	TXDRDY			Enable or disable interrupt for event TXDRDY
			Disabled	0	Disable
			Enabled	1	Enable
F	RW	ENDTX			Enable or disable interrupt for event ENDTX
			Disabled	0	Disable
			Enabled	1	Enable
G	RW	ERROR			Enable or disable interrupt for event ERROR
			Disabled	0	Disable
			Enabled	1	Enable
H	RW	RXTO			Enable or disable interrupt for event RXTO
			Disabled	0	Disable
			Enabled	1	Enable
I	RW	RXSTARTED			Enable or disable interrupt for event RXSTARTED
			Disabled	0	Disable
			Enabled	1	Enable
J	RW	TXSTARTED			Enable or disable interrupt for event TXSTARTED

Bit number	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
ID										L		J	I		H								G	F	E			D		C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			Disabled	0	Disable
			Enabled	1	Enable
L	RW	TXSTOPPED			Enable or disable interrupt for event TXSTOPPED
			Disabled	0	Disable
			Enabled	1	Enable

6.19.9.35 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID										L		J	I		H								G	F	E			D		C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CTS			Write '1' to enable interrupt for event CTS
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	NCTS			Write '1' to enable interrupt for event NCTS
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	RXDRDY			Write '1' to enable interrupt for event RXDRDY
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
D	RW	ENDRX			Write '1' to enable interrupt for event ENDRX
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
E	RW	TXDRDY			Write '1' to enable interrupt for event TXDRDY
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
F	RW	ENDTX			Write '1' to enable interrupt for event ENDTX
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
G	RW	ERROR			Write '1' to enable interrupt for event ERROR
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
H	RW	RXTO			Write '1' to enable interrupt for event RXTO
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
I	RW	RXSTARTED			Write '1' to enable interrupt for event RXSTARTED
			Set	1	Enable

Bit number	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
ID										L		J	I		H								G	F	E			D		C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
J	RW	TXSTARTED			Write '1' to enable interrupt for event TXSTARTED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
L	RW	TXSTOPPED			Write '1' to enable interrupt for event TXSTOPPED
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.19.9.36 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID										L		J	I		H								G	F	E			D		C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CTS			Write '1' to disable interrupt for event CTS
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
B	RW	NCTS			Write '1' to disable interrupt for event NCTS
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
C	RW	RXDRDY			Write '1' to disable interrupt for event RXDRDY
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
D	RW	ENDRX			Write '1' to disable interrupt for event ENDRX
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
E	RW	TXDRDY			Write '1' to disable interrupt for event TXDRDY
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
F	RW	ENDTX			Write '1' to disable interrupt for event ENDTX
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
G	RW	ERROR			Write '1' to disable interrupt for event ERROR
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
H	RW	RXTO			Write '1' to disable interrupt for event RXTO

Bit number	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
ID										L		J	I		H								G	F	E			D		C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
I	RW	RXSTARTED			Write '1' to disable interrupt for event RXSTARTED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
J	RW	TXSTARTED			Write '1' to disable interrupt for event TXSTARTED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled
L	RW	TXSTOPPED			Write '1' to disable interrupt for event TXSTOPPED
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.19.9.37 ERRORSRC

Address offset: 0x480

Error source

Note : this register is read / write one to clear.

Bit number	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
ID																													D	C	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	OVERRUN			Overrun error A start bit is received while the previous data still lies in RXD. (Previous data is lost.)
			NotPresent	0	Read: error not present
			Present	1	Read: error present
B	RW	PARITY			Parity error A character with bad parity is received, if HW parity check is enabled.
			NotPresent	0	Read: error not present
			Present	1	Read: error present
C	RW	FRAMING			Framing error occurred A valid stop bit is not detected on the serial data input after all bits in a character have been received.
			NotPresent	0	Read: error not present
			Present	1	Read: error present
D	RW	BREAK			Break condition The serial data input is '0' for longer than the length of a data frame. (The data frame length is 10 bits without parity bit, and 11 bits with parity bit.).
			NotPresent	0	Read: error not present
			Present	1	Read: error present

6.19.9.38 ENABLE

Address offset: 0x500

Enable UART

Bit number	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
ID																													A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ENABLE			Enable or disable UARTE
			Disabled	0	Disable UARTE
			Enabled	8	Enable UARTE

6.19.9.39 PSEL.RTS

Address offset: 0x508

Pin select for RTS signal

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.19.9.40 PSEL.TXD

Address offset: 0x50C

Pin select for TXD signal

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.19.9.41 PSEL.CTS

Address offset: 0x510

Pin select for CTS signal

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.19.9.42 PSEL.RXD

Address offset: 0x514

Pin select for RXD signal

Bit number	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
ID	C																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN		[0..31]	Pin number
C	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

6.19.9.43 BAUDRATE

Address offset: 0x524

Baud rate. Accuracy depends on the HFCLK source selected.

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x04000000	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	BAUDRATE			Baud rate
			Baud1200	0x0004F000	1200 baud (actual rate: 1205)
			Baud2400	0x0009D000	2400 baud (actual rate: 2396)
			Baud4800	0x0013B000	4800 baud (actual rate: 4808)
			Baud9600	0x00275000	9600 baud (actual rate: 9598)
			Baud14400	0x003AF000	14400 baud (actual rate: 14401)
			Baud19200	0x004EA000	19200 baud (actual rate: 19208)
			Baud28800	0x0075C000	28800 baud (actual rate: 28777)
			Baud31250	0x00800000	31250 baud
			Baud38400	0x009D0000	38400 baud (actual rate: 38369)
			Baud56000	0x00E50000	56000 baud (actual rate: 55944)
			Baud57600	0x00EB0000	57600 baud (actual rate: 57554)
			Baud76800	0x013A9000	76800 baud (actual rate: 76923)
			Baud115200	0x01D60000	115200 baud (actual rate: 115108)
			Baud230400	0x03B00000	230400 baud (actual rate: 231884)
			Baud250000	0x04000000	250000 baud
			Baud460800	0x07400000	460800 baud (actual rate: 457143)
			Baud921600	0x0F000000	921600 baud (actual rate: 941176)
			Baud1M	0x10000000	1Mega baud

6.19.9.44 RXD.PTR

Address offset: 0x534

Data pointer

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			Data pointer

Note: See the memory chapter for details about which memories are available for EasyDMA.

6.19.9.45 RXD.MAXCNT

Address offset: 0x538

Maximum number of bytes in receive buffer

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXCNT		[1..0x1FFF]	Maximum number of bytes in receive buffer

6.19.9.46 RXD.AMOUNT

Address offset: 0x53C

Number of bytes transferred in the last transaction

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	AMOUNT		[1..0x1FFF]	Number of bytes transferred in the last transaction

6.19.9.47 TXD.PTR

Address offset: 0x544

Data pointer

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	PTR			Data pointer

Note: See the memory chapter for details about which memories are available for EasyDMA.

6.19.9.48 TXD.MAXCNT

Address offset: 0x548

Maximum number of bytes in transmit buffer

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	MAXCNT		[1..0x1FFF]	Maximum number of bytes in transmit buffer

6.19.9.49 TXD.AMOUNT

Address offset: 0x54C

Number of bytes transferred in the last transaction

Bit number	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
ID																				A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	AMOUNT		[1..0x1FFF]	Number of bytes transferred in the last transaction

6.19.9.50 CONFIG

Address offset: 0x56C

Configuration of parity and hardware flow control

Bit number	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
ID																												C	B	B	B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	HWFC			Hardware flow control
			Disabled	0	Disabled
			Enabled	1	Enabled
B	RW	PARITY			Parity
			Excluded	0x0	Exclude parity bit
			Included	0x7	Include even parity bit
C	RW	STOP			Stop bits
			One	0	One stop bit
			Two	1	Two stop bits

6.19.10 Electrical specification

6.19.10.1 UARTE electrical specification

Symbol	Description	Min.	Typ.	Max.	Units
f_{UARTE}	Baud rate for UARTE[16].			1000	kbps
$t_{UARTE,CTSH}$	CTS high time	1			µs
$t_{UARTE,START}$	Time from STARTRX/STARTTX task to transmission started	..	..	..	µs

16 High baud rates may require GPIOs to be set as High Drive, see GPIO chapter for more details.

6.20 WDT — Watchdog timer

A countdown watchdog timer using the low-frequency clock source (LFCLK) offers configurable and robust protection against application lock-up.

The watchdog timer is started by triggering the START task.

The watchdog can be paused during long CPU sleep periods for low power applications and when the debugger has halted the CPU. The watchdog is implemented as a down-counter that generates a TIMEOUT event when it wraps over after counting down to 0. When the watchdog timer is started through the START task, the watchdog counter is loaded with the value specified in the CRV register. This counter is also reloaded with the value specified in the CRV register when a reload request is granted.

The watchdog's timeout period is given by:

```
timeout [s] = ( CRV + 1 ) / 32768
```

When started, the watchdog will automatically force the 32.768 kHz RC oscillator on as long as no other 32.768 kHz clock source is running and generating the 32.768 kHz system clock, see chapter CLOCK — Clock control on page 65.

6.20.1 Reload criteria

The watchdog has eight separate reload request registers, which shall be used to request the watchdog to reload its counter with the value specified in the CRV register. To reload the watchdog counter, the special value 0x6E524635 needs to be written to all enabled reload registers.

One or more RR registers can be individually enabled through the RREN register.

6.20.2 Temporarily pausing the watchdog

By default, the watchdog will be active counting down the down-counter while the CPU is sleeping and when it is halted by the debugger. It is however possible to configure the watchdog to automatically pause while the CPU is sleeping as well as when it is halted by the debugger.

6.20.3 Watchdog reset

A TIMEOUT event will automatically lead to a watchdog reset.

See Reset on page 54 for more information about reset sources. If the watchdog is configured to generate an interrupt on the TIMEOUT event, the watchdog reset will be postponed with two 32.768 kHz clock cycles after the TIMEOUT event has been generated. Once the TIMEOUT event has been generated, the impending watchdog reset will always be effectuated.

The watchdog must be configured before it is started. After it is started, the watchdog's configuration registers, which comprise registers CRV, RREN, and CONFIG, will be blocked for further configuration.

The watchdog can be reset from several reset sources, see Reset behavior on page 55.

When the device starts running again, after a reset, or waking up from OFF mode, the watchdog configuration registers will be available for configuration again.

6.20.4 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50018000 0x40018000	WDT	WDT : S WDT : NS	US	NA	Watchdog timer	

Table 100: Instances

Register	Offset	Security	Description
TASKS_START	0x000		Start the watchdog
SUBSCRIBE_START	0x080		Subscribe configuration for task START
EVENTS_TIMEOUT	0x100		Watchdog timeout
PUBLISH_TIMEOUT	0x180		Publish configuration for event TIMEOUT
INTENSET	0x304		Enable interrupt
INTENCLR	0x308		Disable interrupt
RUNSTATUS	0x400		Run status
REQSTATUS	0x404		Request status
CRV	0x504		Counter reload value
RREN	0x508		Enable register for reload request registers
CONFIG	0x50C		Configuration register
RR[0]	0x600		Reload request 0
RR[1]	0x604		Reload request 1
RR[2]	0x608		Reload request 2
RR[3]	0x60C		Reload request 3
RR[4]	0x610		Reload request 4
RR[5]	0x614		Reload request 5
RR[6]	0x618		Reload request 6
RR[7]	0x61C		Reload request 7

Table 101: Register overview

6.20.4.1 TASKS_START

Address offset: 0x000

Start the watchdog

Bit number	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
ID																																A
Reset 0x00000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ID	Acce	Field	Value ID	Value	Description
A	W	TASKS_START			Start the watchdog
			Trigger	1	Trigger task

6.20.4.2 SUBSCRIBE_START

Address offset: 0x080

Subscribe configuration for task START

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that task START will subscribe to
B	RW	EN			
			Disabled	0	Disable subscription
			Enabled	1	Enable subscription

6.20.4.3 EVENTS_TIMEOUT

Address offset: 0x100

Watchdog timeout

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	EVENTS_TIMEOUT			Watchdog timeout
			NotGenerated	0	Event not generated
			Generated	1	Event generated

6.20.4.4 PUBLISH_TIMEOUT

Address offset: 0x180

Publish configuration for event TIMEOUT

Bit number	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
ID	B																												A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	CHIDX		[15..0]	Channel that event TIMEOUT will publish to.
B	RW	EN			
			Disabled	0	Disable publishing
			Enabled	1	Enable publishing

6.20.4.5 INTENSET

Address offset: 0x304

Enable interrupt

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	TIMEOUT			Write '1' to enable interrupt for event TIMEOUT
			Set	1	Enable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.20.4.6 INTENCLR

Address offset: 0x308

Disable interrupt

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	TIMEOUT			Write '1' to disable interrupt for event TIMEOUT
			Clear	1	Disable
			Disabled	0	Read: Disabled
			Enabled	1	Read: Enabled

6.20.4.7 RUNSTATUS

Address offset: 0x400

Run status

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	RUNSTATUSWDT			Indicates whether or not the watchdog is running
			NotRunning	0	Watchdog not running
			Running	1	Watchdog is running

6.20.4.8 REQSTATUS

Address offset: 0x404

Request status

Bit number	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
ID																									H	G	F	E	D	C	B	A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A-H	R	RR[i] (i=0..7)			Request status for RR[i] register
			DisabledOrRequested	0	RR[i] register is not enabled, or are already requesting reload
			EnabledAndUnrequested	1	RR[i] register is enabled, and are not yet requesting reload

6.20.4.9 CRV

Address offset: 0x504

Counter reload value

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	CRV		[0x0000000F..0xFFFFFFFF]	Counter reload value in number of cycles of the 32.768 kHz clock

6.20.4.10 RREN

Address offset: 0x508

Enable register for reload request registers

Bit number	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
ID																									H	G	F	E	D	C	B	A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A-H	RW	RR[i] (i=0..7)			Enable or disable RR[i] register
			Disabled	0	Disable RR[i] register
			Enabled	1	Enable RR[i] register

6.20.4.11 CONFIG

Address offset: 0x50C

Configuration register

Bit number	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
ID																													C			A
Reset 0x00000001	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	SLEEP			Configure the watchdog to either be paused, or kept running, while the CPU is sleeping
			Pause	0	Pause watchdog while the CPU is sleeping
			Run	1	Keep the watchdog running while the CPU is sleeping
C	RW	HALT			Configure the watchdog to either be paused, or kept running, while the CPU is halted by the debugger
			Pause	0	Pause watchdog while the CPU is halted by the debugger
			Run	1	Keep the watchdog running while the CPU is halted by the debugger

6.20.4.12 RR[n] (n=0..7)

Address offset: 0x600 + (n × 0x4)

Reload request n

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	RR			Reload request register
			Reload	0x6E524635	Value to request a reload of the watchdog timer

6.20.5 Electrical specification

6.20.5.1 Watchdog Timer Electrical Specification

Symbol	Description	Min.	Typ.	Max.	Units
t_{WDT}	Time out interval	31 µs		36 h	

7 LTE modem

7.1 Introduction

The long term evolution (LTE) modem consists of baseband processing and RF parts, which together implement a complete 3GPP LTE release 13 (Rel-13) Cat-M1 and Cat-NB1 and LTE release 14 (Rel-14) Cat-NB1 and Cat-NB2 capable product.

As illustrated in the image below, the following is a part of the LTE modem:

- RF transceiver
- Modem baseband (BB)
- Embedded flash/RAM
- Modem host processor and peripherals

The modem baseband and host processor provide functions for the LTE L1, L2 and L3 (layer 1, 2 and 3 respectively) as well as IP communication layers. Modem peripherals provide hardware services for modem operating system and for modem secure execution environment.

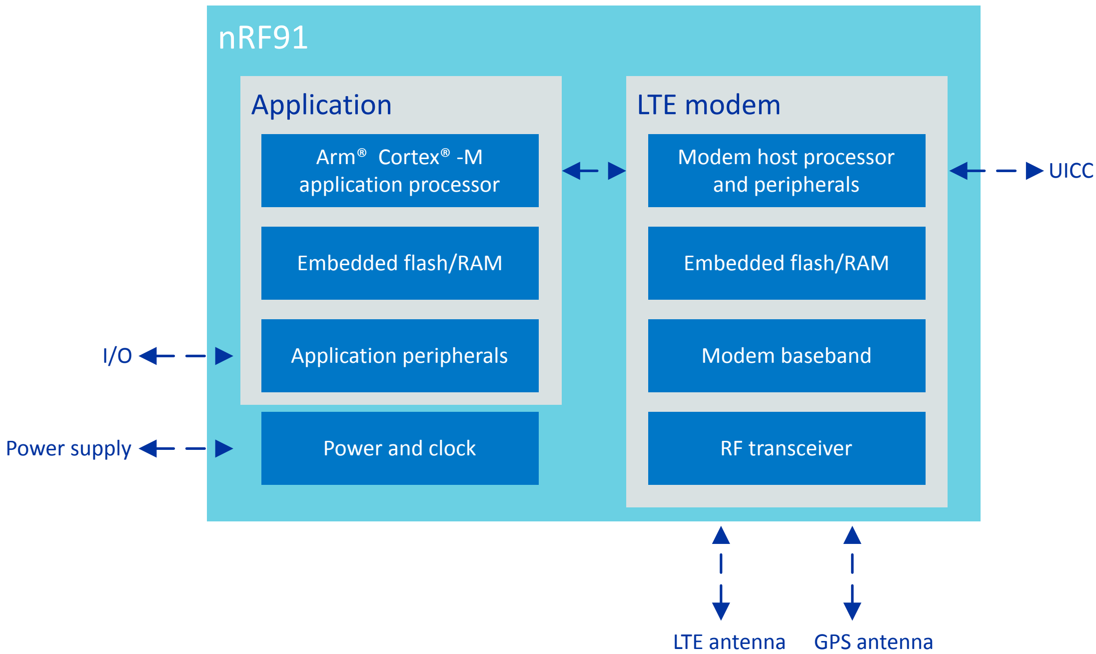

Figure 108: LTE modem within the nRF91

Application and modem domains are interacting through interprocessor communication (IPC) mechanism. LTE modem is accessible to user through the modem API.

The application processor is the master in the system and responsible for starting and stopping of the modem. LTE modem enables the clocks and power required for its own operation. Shared resources, such as e.g. clocks, are handled within the platform and require no user involvement. In cases where a hard fault is detected in the modem, the application domain will perform a hard reset for the modem.

> **Note:** For details regarding the modem API, please refer to *nRF Connect SDK* document and *nRF91 AT Commands, Command Reference Guide* document.

Key features of the LTE modem are:

- Complete modem with baseband and RF transceiver
- 3GPP release 13 compliant LTE categories:
  - Cat-M1 (eMTC - enhanced machine type communication)
  - Cat-NB1 (NB-IoT - narrowband internet of things (IoT))
- 3GPP release 14 compliant LTE categories:
  - Cat-NB1 (NB-IoT)
  - Cat-NB2 (NB-IoT)
- Power saving modes
- Supporting LTE bands from 700 MHz to 2.2 GHz through a single typical 50 Ω antenna pin.
- RX sensitivity: -108 dBm for Cat-M1 and -114 dBm for Cat-NB1 and Cat-NB2
  - As defined in 3GPP conformance test specification TS 36.521-1
- 1.8 V MIPI RFFE (RF front-end) digital control interface and MAGPIO control interface for external RF applications.
- 1.8 V UICC (universal integrated circuit card) interface, based on ISO/IEC 7816-3 and compliant with:
  - ICC (ETSI TS 102 221)
  - eUICC (ETSI TS 103 383)

> **Note:** nRF9160 is able to run different modem FW builds that define the final modem feature set in a specific nRF9160 based application.

7.2 SIM card interface

LTE modem supports the UICC (universal integrated circuit card) interface.

Only the UICCs with the electrical interface specified in ISO/IEC 7816-3 are supported, meaning that the UICCs with IC-USB, CLF or MMC interfaces are not supported.

The supported UICC/eUICC interface is compliant with:

- ETSI TS 102 221: Smart Cards; UICC-Terminal interface; Physical and logical characteristics
- ETSI TS 103 383: Smart Cards; Embedded UICC; Requirements Specification

The physical interface towards the eUICC is the same as towards the removable UICC.

Only the class C (supply voltage 1.8 V nominal) operation is supported. Support for the legacy class B (supply voltage 3.0 V nominal) operation must be built with external components, including the external power supply and the level shifters towards the LTE modem UICC interface.

LTE modem controls the physical interfaces towards the UICC and implements the transport protocol over the four-pin ISO/IEC 7816-3 interface:

- VCC (power supply): LTE modem drives this
- CLK (clock signal): LTE modem drives this
- RST (reset signal): LTE modem drives this
- I/O (input/output serial data): Bi-directional

The interface and the connections between LTE modem, UICC connector, and the ESD device is shown in the figure below.

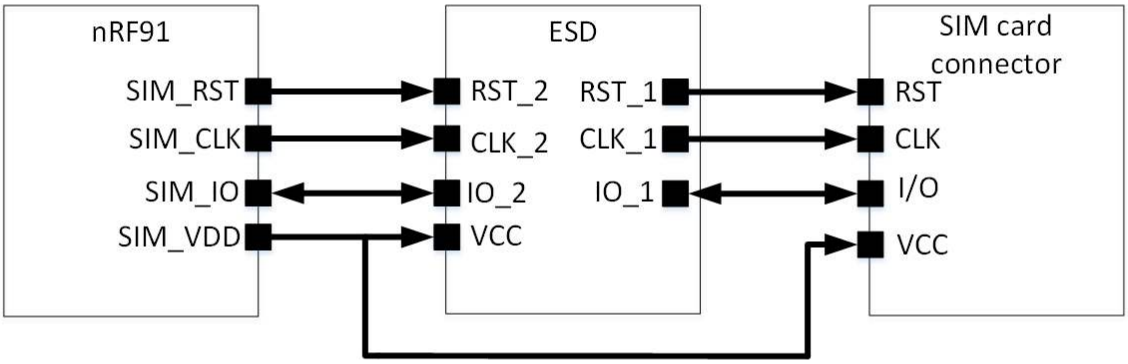

Figure 109: Connections between LTE modem, card connector, and the ESD device

Only standard transmission speeds are supported as specified in ETSI TS 102 221.

Important: LTE modem must be stopped through the modem API, before removing the UICC.

An ESD (electrostatic discharge) protection device compatible with UICC cards must be used between the removable card and the LTE modem, to protect LTE modem against a harmful electrostatic discharge from the card connector.

7.3 LTE modem coexistence interface

LTE modem uses a dedicated three-pin interface for RF interference avoidance towards a companion radio device e.g. an external positioning device or Bluetooth® Low Energy device.

The inputs and outputs for this interface:

- COEX0: Input to the LTE modem from the external device. When active high, indicates that the external device transceiver is turned on.
  - When internal GPS is used, COEX0 can be used as active high control for the external LNA component.
- COEX1: Output from the LTE modem to the external device. Active high time mark pulse, which is synchronous to LTE system time.
  - When internal GPS is used, COEX1 delivers the GPS 1PPS (one pulse per second) time mark pulse.
- COEX2: Output from the LTE modem to the external device. When active high, indicates that the LTE modem transceiver is turned on.
  - COEX2 can also be treated as active low grant from LTE modem to the external device, indicating grant to transmit and receive.

Note: COEX2 pin requires an external pull-down resistor in 100 kΩ size range to be used.

7.4 LTE modem RF control external interface

LTE modem provides dedicated 1.8 V digital interfaces for controlling external RF applications, such as antenna tuner devices:

- MIPI RFFE interface pins: VIO, SCLK, SDATA.
- MAGPIO interface pins: MAGPIO0, MAGPIO1, MAGPIO2.

LTE modem drives these outputs timing accurately according to LTE protocol timing to set e.g. the correct antenna tuner settings per used frequency. User needs to inform the LTE modem through the modem API about the particular RF application e.g. antenna tuner device configuration, so that LTE modem knows how to drive it.

Note: For details regarding the modem API and supported RF external control features, please refer to *nRF91 AT Commands, Command Reference Guide* document.

7.5 RF front-end interface

nRF9160 has a single-ended (SE) 50 Ω antenna interface to connect directly to antenna.

7.6 Electrical specification

7.6.1 Key RF parameters for Cat-M1

Note: For certification status, please refer to Regulatory information on page 398.

Symbol	Description	Min.	Typ.	Max.	Units
Supported LTE	Supported LTE standards		LTE Rel-13 Cat-M1 HD-FDD		
Bands supported	Bands supported		B1, B2, B3, B4, B5, B8, B12, B13, B14, B17, B20, B25, B26, B28, B66		
Transmission bandwidth	Maximum bandwidth		1.4		MHz

7.6.2 Key RF parameters for Cat-NB1 and Cat-NB2

Note: For certification status, please refer to Regulatory information on page 398.

Note: There is no foreseen NB-IoT network deployment for FCC bands closer than 200 kHz from band edge, hence our device will not transmit in FCC bands on channels that are closer than 200kHz to band edge.

Symbol	Description	Min.	Typ.	Max.	Units
Supported LTE	Supported LTE standards		LTE Rel-13 Cat-NB1 HD-FDD, LTE Rel-14 Cat-NB1 and Cat-NB2 HD-FDD		
Bands supported	Bands supported		B1, B2, B3, B4, B5, B8, B12, B13, B17, B20, B25, B26, B28, B66		
Transmission bandwidth	Maximum bandwidth		200		kHz

7.6.3 Receiver parameters for Cat-M1

Symbol	Description	Min.	Typ.	Max.	Units
$Freq_{range\_ANT\_RX}$	RX operation frequency range at ANT (pin 61)	729		2200	MHz
Z_{in}	Input impedance, single-ended		50		Ω
Sensitivity, low band	LTE 1.4 MHz without coverage extension	-103	-108		dBm
Sensitivity, mid band	LTE 1.4 MHz without coverage extension	-103	-107		dBm

7.6.4 Receiver parameters for Cat-NB1 and Cat-NB2

Symbol	Description	Min.	Typ.	Max.	Units
$Freq_{range\_ANT\_RX}$	RX operation frequency range at ANT (pin 61)	729		2200	MHz
Z_{in}	Input impedance, single-ended		50		Ω
Sensitivity, low band	NB 200 kHz without coverage extension	-108	-114		dBm
Sensitivity, mid band	NB 200 kHz without coverage extension	-108	-113		dBm

7.6.5 Transmitter parameters for Cat-M1

Symbol	Description	Min.	Typ.	Max.	Units
$Freq_{range\_ANT\_TX}$	TX operation frequency range at ANT (pin 61)	699		1980	MHz
Z_{out}	Output impedance, single-ended		50		Ω
Maximum output power	Maximum output power		23		dBm
Minimum output power	Minimum output power		-40		dBm
Pout maximum accuracy	Pout maximum accuracy		+-2		dB

7.6.6 Transmitter parameters for Cat-NB1 and Cat-NB2

Symbol	Description	Min.	Typ.	Max.	Units
$Freq_{range\_ANT\_TX}$	TX operation frequency range at ANT (pin 61)	699		1980	MHz
Z_{out}	Output impedance, single-ended		50		Ω
Maximum output power	Maximum output power		23		dBm
Minimum output power	Minimum output power		-40		dBm
Pout maximum accuracy	Pout maximum accuracy		+-2		dB

8 GPS receiver

The GPS receiver supports GPS L1 C/A reception. Operation is time multiplexed with LTE modem, and it is possible to obtain the GPS position either while the LTE is in RRC Idle mode or power saving mode (PSM), or when the LTE modem is completely deactivated.

The application processor is the master in the system and responsible for starting and stopping of the GPS receiver. GPS receiver is accessible to user through the GPS API.

Note: For details regarding the modem API, please refer to *nRF Connect SDK* document and *nRF91 AT Commands, Command Reference Guide* document.

Key features of the GPS receiver are:

- GPS L1 C/A supported
- Optimized for low-power and low-cost IoT applications
- Modes of operation:
  - Single shot (cold start mode by default)
  - Position fix per fixed interval, configurable between 10 s - 0.5 hour: first start with cold start, sequential fixes with hot start
  - Continuous tracking
- Power saving mode:
  - Duty-cycled continuous tracking operation
- Antenna interface:
  - External low-noise amplifier (LNA) with SAW filter recommended on the GPS antenna input
  - Dedicated GPS antenna, or shared antenna with LTE

Note: There must be minumum 27dB attenuation to out of band power to avoid blocking high power RF signals to GPS receiver input. This can be achieved for example by a SAW filter at the output of the external LNA.

8.1 Electrical specification

Table below summarises GPS receiver performance parameters.

Symbol	Description	Value	Unit
Sensitivity, cold	Acquisition sensitivity, cold start	-142	dBm
Sensitivity, hot	Acquisition sensitivity, hot start	-145	dBm
Sensitivity, tracking	Tracking sensitivity	-151	dBm
TTFF, cold	Acquisition time (time to first fix (TTFF)), cold start, open sky, typical	36	s
TTFF, hot	Acquisition time (TTFF), hot start, open sky, typical	1.3	s
Accuracy, periodic	Positioning accuracy (CEP50), periodic tracking	5	m
Accuracy, continuous	Positioning accuracy (CEP50), continuous tracking	3	m

Table 102: GPS electrical specification

9 Debug and trace

9.1 Overview

The debug and trace system offers a flexible and powerful mechanism for non-intrusive debugging.

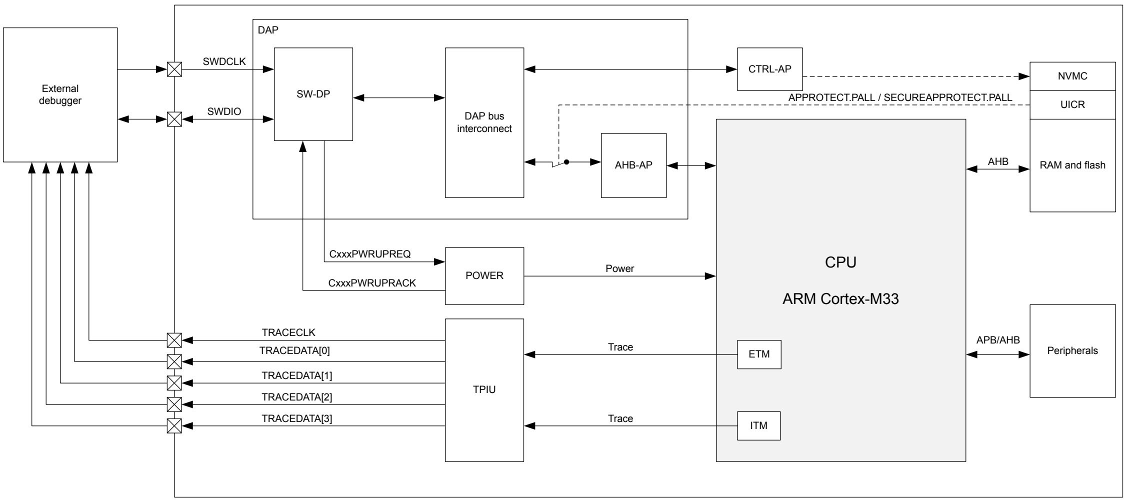

Figure 110: Debug and trace overview

The main features of the debug and trace system include:

- Two-pin serial wire debug (SWD) interface, protocol version 1
- Access port connection
  - Breakpoint unit (BPU) supports eight hardware breakpoint comparators
  - Data watchpoint and trace (DWT) unit supports four watchpoint comparators
  - Instrumentation trace macrocell (ITM)
  - Embedded trace macrocell (ETM)
  - Access protection through APPROTECT, ERASEPROTECT and SECUREAPPROTECT
- Trace port interface unit (TPIU)
  - 4-bit parallel trace of ITM and ETM trace data

Note: When a system contains multiple CPU domains, it is important to notice that if one domain (subsystem A) has master rights on another domain (subsystem B), the master subsystem can have access to data from the slave subsytem. In this example, even if subsystem B is locked by APPROTECT or ERASEPROTECT, subsystem A can access some data for subsystem B. Consequently, even if the security permissions are managed per subsystem, it is mandatory to have a global approach to the protection. Protecting a slave subsystem does not guarantee system security if the master subsystem is not protected.

9.1.1 Special consideration regarding debugger access

A debugger, if desired, can be restricted to debug non-secure code only, and access non-secure memory regions and peripherals using register SECUREAPPROTECT on page 42. Register APPROTECT on page 41 will block all debugger access.

Debugger accesses are controlled as described in table below.

Debugging capability	UICR.APPROTECT.PALL	UICR.SECUREAPPROTECT.PALL
Secure and non-secure code	Unprotected	Unprotected
Non-secure code only	Unprotected	Protected
No debugging possible	Protected	-

Table 103: Debugger access control

If a RAM or flash region has its permission set to allow code execution, the content of this region will be visible to the debugger even if the read permission is not set. This allows a debugger to display the content of the code being executed. For more about how to configure permissions, please refer to SPU - System protection unit on page 260.

9.1.2 DAP - Debug access port

An external debugger can access the device via the debug access port (DAP).

The DAP implements a standard ARM® CoreSight™ serial wire debug port (SW-DP). The SW-DP implements the serial wire debug (SWD) protocol that is a two-pin serial interface, see SWDCLK and SWDIO illustrated in figure Debug and trace overview on page 371.

In addition to the default access port in the application CPU (AHB-AP), the DAP includes a custom control access port (CTRL-AP), described in more detail in CTRL-AP - Control access port on page 374.

Note:

- The SWDIO line has an internal pull-up resistor.
- The SWDCLK line has an internal pull-down resistor.

There are several access ports that connect to different parts of the system. An overview is given in the table below.

AP ID	Type	Description
0	AHB-AP	Application subsystem access port
3	APB-AP	CoreSight™ subsystem access port
4	CTRL-AP	Application subsystem control access port

Table 104: Access port overview

The standard ARM® components are documented in *ARM CoreSight SoC-400 Technical Reference Manual, revision r3p2*. The control access port (CTRL-AP) is proprietary, and described in more detail in CTRL-AP - Control access port on page 374.

9.1.3 Debug interface mode

Before the external debugger can access the CPU's access port (AHB-AP) or the control access port (CTRL-AP), the debugger must first request the device to power up via CxxxPWRUPREQ in the SWJ-DP.

As long as the debugger is requesting power via CxxxPWRUPREQ, the device will be in debug interface mode. Otherwise, the device is in normal mode. When a debug session is over, the external debugger must make sure to put the device back into normal mode and then a pin reset should be performed. The reason is that the overall power consumption is higher in debug interface mode compared to normal mode.

Some peripherals behave differently in debug interface mode compared to normal mode. The differences are described in more detail in the chapters of the affected peripherals.

For details on how to use the debug capabilities, please read the debug documentation of your IDE.

If the device is in System OFF when power is requested via CxxxPWRUPREQ, the system will wake up and the DIF flag in RESETREAS on page 64 will be set.

9.1.4 Real-time debug

The device supports real-time debugging, which allows interrupts to execute to completion in real time when breakpoints are set in thread mode or lower priority interrupts.

Real-time debugging thus enables the developer to set a breakpoint and single-step through their code without a failure of the real-time event-driven threads running at higher priority. For example, this enables the device to continue to service the high-priority interrupts of an external controller or sensor without failure or loss of state synchronization while the developer steps through code in a low-priority thread.

9.1.5 Trace

The device supports ETM and ITM trace.

Trace data from the ETM and the ITM is sent to an external debugger via a 4-bit wide parallel trace port (TPIU), see TRACEDATA[0] through TRACEDATA[3], and TRACECLK in Debug and trace overview on page 371.

For details on how to use the trace capabilities, please read the debug documentation of your IDE.

TPIU's trace pins are multiplexed with GPIOs, see Pin assignments on page 386 for more information.

> **Note:** To configure the trace data delivery to the device trace port, use the MDK system start-up file included as of MDK version 8.26.0.

Trace speed is configured in the TRACEPORTSPEED (Retained) on page 385 register. The speed of the trace pins depends on the DRIVE setting of the GPIOs that the trace pins are multiplexed with. See GPIO — General purpose input/output on page 95 for information about how to set drive settings. Only S0S1 and H0H1 drives are suitable for debugging. S0S1 is the default DRIVE at reset. If parallel or serial trace port signals are not fast enough in the debugging conditions, all GPIOs in use for tracing should be set to high drive (H0H1). The user shall make sure that DRIVE setting for these GPIOs is not overwritten by software during the debugging session.

9.1.6 Registers

Register	Offset	Security	Description
TARGETID	0x042		The TARGETID register provides information about the target when the host is connected to a single device. The TARGETID register is accessed by a read of DP register 0x4 when the DPBANKSEL bit in the SELECT register is set to 0x2.

Table 105: Register overview

9.1.6.1 TARGETID

Address offset: 0x042

The TARGETID register provides information about the target when the host is connected to a single device.

The TARGETID register is accessed by a read of DP register 0x4 when the DPBANKSEL bit in the SELECT register is set to 0x2.

Bit number	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
ID	D	D	D	D	C	C	C	C	C	C	C	C	C	C	C	C					B	B	B	B	B	B	B	B	B	B	B	A
Reset 0x10090289	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**0**	**1**

ID	Acce	Field	Value ID	Value	Description
A	R	UNUSED			Reserved, read-as-one
B	R	TDESIGNER			An 11-bit code: JEDEC JEP106 continuation code and identity code. The ID identifies the designer of the part.
			NordicSemi	0x144	Nordic Semiconductor ASA
C	R	TPARTNO			Part number
			nRF91	9	nRF91 Series
D	R	TREVISION			Target revision
			nRF9160	1	nRF9160

9.1.7 Electrical specification

9.1.7.1 Trace port

Symbol	Description	Min.	Typ.	Max.	Units
T_{cyc}	Clock period, as defined by ARM (See ARM Infocenter, Embedded Trace Macrocell Architecture Specification, Trace Port Physical Interface, Timing specifications)	..	..	..	ns

9.2 CTRL-AP - Control access port

The control access port (CTRL-AP) is a custom access port that enables control of the device when other debug access ports (DAP) have been disabled by the access port protection.

For an overview of the other debug access ports, see DAP - Debug access port on page 372.

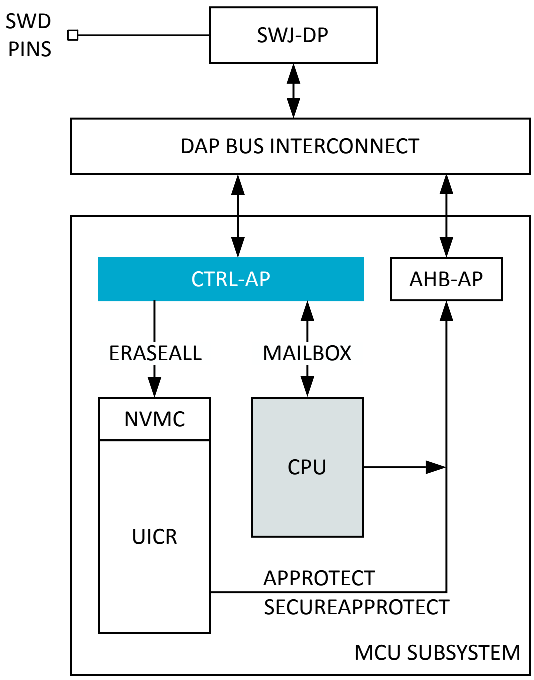

Figure 111: Control access port details

Access port protection (APPROTECT) blocks the debugger access to the AHB-AP, and prevents read and write access to all CPU registers and memory-mapped addresses. It is possible to enable access port protection for both secure and non-secure mode, using registers UICR.SECUREAPPROTECT and UICR.APPROTECT respectively. The debugger can use register APPROTECT.STATUS on page 378 to read the status of secure and non-secure access port protection.

CTRL-AP has the following features:

- Soft reset
- Erase all
- Mailbox interface
- Debug of protected devices

9.2.1 Reset request

The debugger can request the device to perform a soft reset.

Register RESET on page 377 is used to request the soft reset. Once the soft reset is performed, the reset reason is accessible to on-chip firmware through register RESETREAS. For more information about the soft reset, see Reset on page 54.

9.2.2 Erase all

Erase all function gives debugger the possibility of triggering an erase of flash, user information configuration registers (UICR), RAM, including all peripheral settings, as well as removing the access port protection.

To trigger an erase all function, the debugger can write to register ERASEALL on page 377. Register ERASEALLSTATUS on page 377 will read as busy for the duration of the operation. After the next reset, the access port protection is removed.

If the debugger performs an erase all function on a slave MCU, the erase sequence will always erase the application MCU first, independently of how the application is protected, before erasing the slave MCU.

Erase all protection

It is possible to prevent debugger from performing an erase all operation by writing to register ERASEPROTECT on page 42. Once the register is configured and the device reset, the CTRL-AP ERASEALL operation is disabled, and all flash write and erase operations are restricted to firmware. In addition, it is still possible to write/erase from debugger as long as APPROTECT on page 41 is not set.

> **Note:** Setting ERASEPROTECT on page 42 has no effect on debugger access, only on erase all operation.

Register ERASEPROTECT.STATUS on page 378 holds the status for erase protection.

9.2.3 Mailbox interface

CTRL-AP implements a mailbox interface which enables the CPU to communicate with a debugger over the SWD interface.

The mailbox interface consists of a transmit register MAILBOX.TXDATA on page 379 with its corresponding status register MAILBOX.TXSTATUS on page 379, and a receive register MAILBOX.RXDATA on page 379 with its corresponding status register MAILBOX.RXSTATUS on page 379. Status bits in registers TXSTATUS/RXSTATUS will be set and cleared automatically when registers TXDATA/RXDATA are written to and read from, independently of the direction.

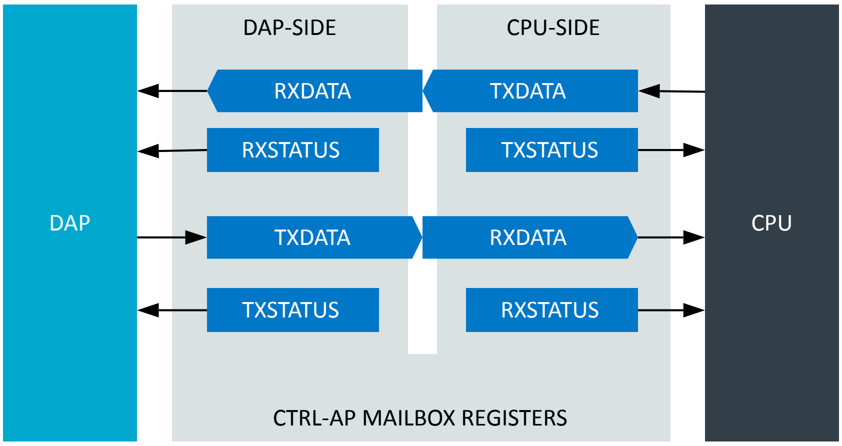

Figure 112: Mailbox register interface

Mailbox transfer sequence

1. Sender writes TXDATA
2. Hardware sets sender's TXSTATUS to DataPending
3. Hardware sets receiver's RXSTATUS to DataPending
4. Receiver reads RXDATA
5. Hardware sets receiver's RXSTATUS to NoDataPending
6. Hardware sets sender's TXSTATUS to NoDataPending

9.2.4 Disabling erase protection

The erase protection mechanism can be disabled in order to return a device to factory default settings upon next reset.

The debugger can read the erase protection status in register ERASEPROTECT.STATUS on page 378.

If ERASEPROTECT has been enabled, both the debugger and on-chip firmware must write the same non-zero 32-bit KEY value into their respective ERASEPROTECT.DISABLE registers in order to disable the erase protection. As soon as both registers have been written with the same non-zero 32-bit KEY value, the device is automatically erased as described in Erase all on page 375. The access ports will be re-enabled on next reset once the secure erase sequence has completed.

Write-once register ERASEPROTECT.LOCK on page 381 should be set to 'Locked' as early as possible in the start-up sequence, preferably as soon as on-chip firmware has determined it does not need to communicate with a debugger over the CTRL-AP mailbox interface. Once written, it will not be possible to remove the erase protection until next reset.

9.2.5 Registers

Register	Offset	Security	Description
RESET	0x000		System reset request.
ERASEALL	0x004		Perform a secure erase of the device, where flash, SRAM and UICR will be erased in sequence. The device will be returned to factory default settings upon next reset.
ERASEALLSTATUS	0x008		Status register for the ERASEALL operation
APPROTECT.STATUS	0x00C		Status register for UICR APPROTECT and SECUREAPPROTECT configuration
ERASEPROTECT.STATUS	0x018		Status register for UICR ERASEPROTECT configuration
ERASEPROTECT.DISABLE	0x01C		Disable ERASEPROTECT and perform ERASEALL
MAILBOX.TXDATA	0x020		Data sent from the debugger to the CPU
MAILBOX.TXSTATUS	0x024		Status to indicate if data sent from the debugger to the CPU has been read

Register	Offset	Security	Description
MAILBOX.RXDATA	0x028		Data sent from the CPU to the debugger
MAILBOX.RXSTATUS	0x02C		Status to indicate if data sent from the CPU to the debugger has been read
IDR	0x0FC		CTRL-AP Identification Register, IDR

Table 106: Register overview

9.2.5.1 RESET

Address offset: 0x000

System reset request.

This register is automatically deactivated by writing Erase to ERASEALL, it is then kept inactive until a reset source affecting the debug system is asserted. See Reset behavior on page 55.

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	RESET			System reset request and status
			NoReset	0	Write to release reset Reading '0' means reset is not active
			Reset	1	Write to hold reset Reading '1' means reset is active

9.2.5.2 ERASEALL

Address offset: 0x004

Perform a secure erase of the device, where flash, SRAM and UICR will be erased in sequence. The device will be returned to factory default settings upon next reset.

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	W	ERASEALL			Return device to factory default settings
			NoOperation	0	No operation
			Erase	1	Erase flash, SRAM and UICR in sequence

9.2.5.3 ERASEALLSTATUS

Address offset: 0x008

Status register for the ERASEALL operation

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	R	ERASEALLSTATUS			Status bit for the ERASEALL operation
			Ready	0	ERASEALL is ready
			Busy	1	ERASEALL is busy (on-going)

9.2.5.4 APPROTECT.STATUS

Address offset: 0x00C

Status register for UICR APPROTECT and SECUREAPPROTECT configuration

Bit number	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
ID																															B	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	APPROTECT			Status bit for access port protection **Note:** Reset value is auto read from the APPROTECT register in UICR
			Enabled	0	APPROTECT is enabled
			Disabled	1	APPROTECT is disabled
B	R	SECUREAPPROTECT			Status bit for secure access port protection **Note:** Reset value is auto read from the SECUREAPPROTECT register in UICR
			Enabled	0	SECUREAPPROTECT is enabled
			Disabled	1	SECUREAPPROTECT is disabled

9.2.5.5 ERASEPROTECT.STATUS

Address offset: 0x018

Status register for UICR ERASEPROTECT configuration

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	PALL			Status bit for erase protection **Note:** Reset value is auto read from the ERASEPROTECT register in UICR
			Enabled	0	ERASEPROTECT is enabled
			Disabled	1	ERASEPROTECT is not enabled and ERASEALL can be performed

9.2.5.6 ERASEPROTECT.DISABLE

Address offset: 0x01C

Disable ERASEPROTECT and perform ERASEALL

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	KEY			The ERASEALL sequence will be initiated if value of KEY fields are non-zero and KEY fields match on both CPU and debugger side

9.2.5.7 MAILBOX.TXDATA

Address offset: 0x020

Data sent from the debugger to the CPU

Writing to this register will automatically set field DataPending in register TXSTATUS

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	Data			Data sent from debugger

9.2.5.8 MAILBOX.TXSTATUS

Address offset: 0x024

Status to indicate if data sent from the debugger to the CPU has been read

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	Status			Status of register DATA
			NoDataPending	0	No data pending in register TXDATA
			DataPending	1	Data pending in register TXDATA

9.2.5.9 MAILBOX.RXDATA

Address offset: 0x028

Data sent from the CPU to the debugger

Reading from this register will automatically set field NoDataPending in register RXSTATUS

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	Data			Data sent from CPU

9.2.5.10 MAILBOX.RXSTATUS

Address offset: 0x02C

Status to indicate if data sent from the CPU to the debugger has been read

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	Status			Status of register DATA
			NoDataPending	0	No data pending in register RXDATA
			DataPending	1	Data pending in register RXDATA

9.2.5.11 IDR

Address offset: 0x0FC

CTRL-AP Identification Register, IDR

Bit number	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
ID	E	E	E	E	D	D	D	D	C	C	C	C	C	C	C	B	B	B	B						A	A	A	A	A	A	A	A
Reset 0x12880000	**0**	**0**	**0**	**1**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	APID			AP Identification
B	R	CLASS			Access Port (AP) class
			NotDefined	0x0	No defined class
			MEMAP	0x8	Memory Access Port
C	R	JEP106ID			JEDEC JEP106 identity code
D	R	JEP106CONT			JEDEC JEP106 continuation code
E	R	REVISION			Revision

9.2.6 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0x50006000	CTRLAPPERI	CTRL_AP_PERI	S	NA	CTRL-AP-PERI	

Table 107: Instances

Register	Offset	Security	Description
MAILBOX.RXDATA	0x400		Data sent from the debugger to the CPU
MAILBOX.RXSTATUS	0x404		Status to indicate if data sent from the debugger to the CPU has been read
MAILBOX.TXDATA	0x480		Data sent from the CPU to the debugger
MAILBOX.TXSTATUS	0x484		Status to indicate if data sent from the CPU to the debugger has been read
ERASEPROTECT.LOCK	0x500		Lock register ERASEPROTECT.DISABLE from being written until next reset
ERASEPROTECT.DISABLE	0x504		Disable ERASEPROTECT and perform ERASEALL

Table 108: Register overview

9.2.6.1 MAILBOX.RXDATA

Address offset: 0x400

Data sent from the debugger to the CPU

Reading from this register will automatically set field NoDataPending in register RXSTATUS

Bit number	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
ID	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	R	RXDATA			Data received from debugger

9.2.6.2 MAILBOX.RXSTATUS

Address offset: 0x404

Status to indicate if data sent from the debugger to the CPU has been read

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	R	RXSTATUS			Status of data in register RXDATA
			NoDataPending	0	No data pending in register RXDATA
			DataPending	1	Data pending in register RXDATA

9.2.6.3 MAILBOX.TXDATA

Address offset: 0x480

Data sent from the CPU to the debugger

Writing to this register will automatically set field DataPending in register TXSTATUS

Bit number	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
ID	A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW	TXDATA			Data sent to debugger

9.2.6.4 MAILBOX.TXSTATUS

Address offset: 0x484

Status to indicate if data sent from the CPU to the debugger has been read

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	R	TXSTATUS			Status of data in register TXDATA
			NoDataPending	0	No data pending in register TXDATA
			DataPending	1	Data pending in register TXDATA

9.2.6.5 ERASEPROTECT.LOCK

Address offset: 0x500

Lock register ERASEPROTECT.DISABLE from being written until next reset

Bit number	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
ID	A
Reset 0x00000000	**0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

ID	Acce	Field	Value ID	Value	Description
A	RW1	LOCK			Lock register ERASEPROTECT.DISABLE from being written until next reset
			Unlocked	0	Register ERASEPROTECT.DISABLE is writeable
			Locked	1	Register ERASEPROTECT.DISABLE is read-only

9.2.6.6 ERASEPROTECT.DISABLE

Address offset: 0x504

Disable ERASEPROTECT and perform ERASEALL

Bit number					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
ID					A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
Reset 0x00000000					**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
ID	Acce	Field	Value ID	Value	Description																															
A	RW	KEY			The ERASEALL sequence will be initiated if value of KEY fields are non-zero and KEY fields match on both CPU and debugger side																															

9.3 TAD - Trace and debug control

A configuration interface for trace and debug is provided so that the CPU can have control over the debug domain.

Since traces are provided via GPIOs, the CPU must configure the corresponding GPIOs accordingly and make sure they are available for a debug connection. The CPU can also control the trace speed, and aquire or release the GPIOs as needed.

9.3.1 Registers

Base address	Peripheral	Instance	Secure mapping	DMA security	Description	Configuration
0xE0080000	TAD	TAD	S	NA	Trace and debug control	

Table 109: Instances

Register	Offset	Security	Description	
CLOCKSTART	0x000		Start all trace and debug clocks.	
CLOCKSTOP	0x004		Stop all trace and debug clocks.	
ENABLE	0x500		Enable debug domain and aquire selected GPIOs	
PSEL.TRACECLK	0x504		Pin configuration for TRACECLK	
PSEL.TRACEDATA0	0x508		Pin configuration for TRACEDATA[0]	
PSEL.TRACEDATA1	0x50C		Pin configuration for TRACEDATA[1]	
PSEL.TRACEDATA2	0x510		Pin configuration for TRACEDATA[2]	
PSEL.TRACEDATA3	0x514		Pin configuration for TRACEDATA[3]	
TRACEPORTSPEED	0x518		Clocking options for the Trace Port debug interface Reset behavior is the same as debug components	Retained

Table 110: Register overview

9.3.1.1 CLOCKSTART

Address offset: 0x000

Start all trace and debug clocks.

Bit number					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
ID																																				A
Reset 0x00000000					**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
ID	Acce	Field	Value ID	Value	Description																															
A	W	START																																		
			Start	1	Start all trace and debug clocks.																															

9.3.1.2 CLOCKSTOP

Address offset: 0x004

Stop all trace and debug clocks.

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	W	STOP			
			Stop	1	Stop all trace and debug clocks.

9.3.1.3 ENABLE

Address offset: 0x500

Enable debug domain and aquire selected GPIOs

Bit number	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
ID																																A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	ENABLE			
			DISABLED	0	Disable debug domain and release selected GPIOs
			ENABLED	1	Enable debug domain and aquire selected GPIOs

9.3.1.4 PSEL.TRACECLK

Address offset: 0x504

Pin configuration for TRACECLK

Bit number	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
ID	B																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN			Pin number
			Traceclk	21	TRACECLK pin **Note:** Only this pin is valid
B	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

9.3.1.5 PSEL.TRACEDATA0

Address offset: 0x508

Pin configuration for TRACEDATA[0]

Bit number	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
ID	B																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN			Pin number
			Tracedata0	22	TRACEDATA0 pin
					Note: Only this pin is valid
B	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

9.3.1.6 PSEL.TRACEDATA1

Address offset: 0x50C

Pin configuration for TRACEDATA[1]

Bit number	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
ID	B																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN			Pin number
			Tracedata1	23	TRACEDATA1 pin
					Note: Only this pin is valid
B	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

9.3.1.7 PSEL.TRACEDATA2

Address offset: 0x510

Pin configuration for TRACEDATA[2]

Bit number	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
ID	B																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN			Pin number
			Tracedata2	24	TRACEDATA2 pin
					Note: Only this pin is valid
B	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

9.3.1.8 PSEL.TRACEDATA3

Address offset: 0x514

Pin configuration for TRACEDATA[3]

Bit number	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
ID	B																											A	A	A	A	A
Reset 0xFFFFFFFF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

ID	Acce	Field	Value ID	Value	Description
A	RW	PIN			Pin number
			Tracedata3	25	TRACEDATA3 pin **Note:** Only this pin is valid
B	RW	CONNECT			Connection
			Disconnected	1	Disconnect
			Connected	0	Connect

9.3.1.9 TRACEPORTSPEED (Retained)

Address offset: 0x518

This register is a retained register

Clocking options for the Trace Port debug interface

Reset behavior is the same as debug components

Bit number	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
ID																															A	A
Reset 0x00000000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ID	Acce	Field	Value ID	Value	Description
A	RW	TRACEPORTSPEED			Speed of Trace Port clock. Note that the TRACECLK pin output will be divided again by two from the Trace Port clock.
			32MHz	0	Trace Port clock is: 32MHz
			16MHz	1	Trace Port clock is: 16MHz
			8MHz	2	Trace Port clock is: 8MHz
			4MHz	3	Trace Port clock is: 4MHz

10 Hardware and layout

10.1 Pin assignments

This section describes the pin assignment and the pin functions.

This device provides flexibility when it comes to routing and configuration of the GPIO pins. However, some pins have recommendations for how the pin should be configured or what it should be used for. See LGA pin assignments on page 386 for more information about this.

10.1.1 Pin assignments

The pin assignment table and figure describe the assignments.

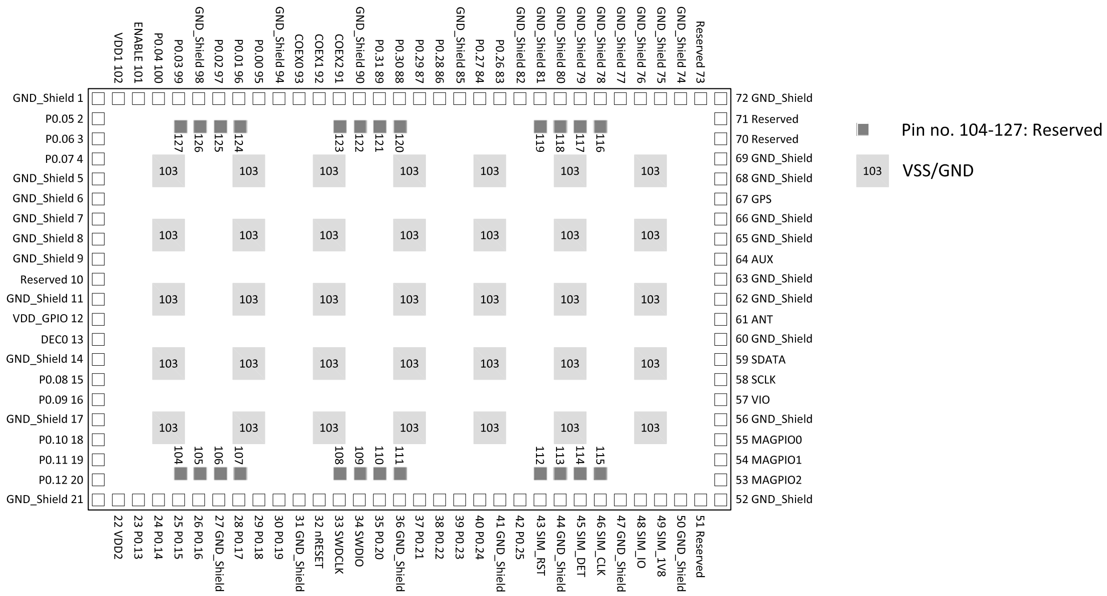

Figure 113: LGA pin assignments, top view

Pin no	Pin name	Function	Description
1	GND_Shield	Power	Ground
2	P0.05	Digital I/O (SoC)	General purpose I/O
3	P0.06	Digital I/O (SoC)	General purpose I/O
4	P0.07	Digital I/O (SoC)	General purpose I/O
5	GND_Shield	Power	Ground
6	GND_Shield	Power	Ground
7	GND_Shield	Power	Ground
8	GND_Shield	Power	Ground
9	GND_Shield	Power	Ground
10	Reserved		Do not connect/reserved for future use
11	GND_Shield	Power	Ground
12	VDD_GPIO	Power	GPIO power supply input and logic level
13	DEC0	Power	Power supply decoupling. Reserved for Nordic use.
14	GND_Shield	Power	Ground
15	P0.08	Digital I/O (SoC)	General purpose I/O
16	P0.09	Digital I/O (SoC)	General purpose I/O

Pin no	Pin name	Function	Description
17	GND_Shield	Power	Ground
18	P0.10	Digital I/O (SoC)	General purpose I/O
19	P0.11	Digital I/O (SoC)	General purpose I/O
20	P0.12	Digital I/O (SoC)	General purpose I/O
21	GND_Shield	Power	Ground
22	VDD2	Power	Supply voltage input
23	P0.13 AIN0	Digital I/O (SoC) Analog input	General purpose I/O. Analog input.
24	P0.14 AIN1	Digital I/O (SoC) Analog input	General purpose I/O. Analog input.
25	P0.15 AIN2	Digital I/O (SoC) Analog input	General purpose I/O. Analog input.
26	P0.16 AIN3	Digital I/O (SoC) Analog input	General purpose I/O. Analog input.
27	GND_Shield	Power	Ground
28	P0.17 AIN4	Digital I/O (SoC) Analog input	General purpose I/O. Analog input.
29	P0.18 AIN5	Digital I/O (SoC) Analog input	General purpose I/O. Analog input.
30	P0.19 AIN6	Digital I/O (SoC) Analog input	General purpose I/O. Analog input.
31	GND_Shield	Power	Ground
32	nRESET	Digital I/O (SoC)	SoC system reset **Note:** External pull-up not allowed.
33	SWDCLK	Digital input	Serial wire debug clock input for debug and programming
34	SWDIO	Digital I/O	Serial wire debug I/O for debug and programming
35	P0.20 AIN7	Digital I/O (SoC) Analog input	General purpose I/O. Analog input.
36	GND_Shield	Power	Ground
37	P0.21 TRACECLK	Digital I/O (SoC) Trace clock	General purpose I/O. Trace buffer clock (optional).
38	P0.22 TRACEDATA0	Digital I/O (SoC) Trace data	General purpose I/O. Trace buffer TRACEDATA[0] (optional).
39	P0.23 TRACEDATA1	Digital I/O (SoC) Trace data	General purpose I/O. Trace buffer TRACEDATA[1] (optional).
40	P0.24 TRACEDATA2	Digital I/O (SoC) Trace data	General purpose I/O. Trace buffer TRACEDATA[2] (optional).
41	GND_Shield	Power	Ground
42	P0.25 TRACEDATA3	Digital I/O (SoC) Trace data	General purpose I/O. Trace buffer TRACEDATA[3] (optional).
43	SIM_RST	Digital I/O (SoC)	SIM reset
44	GND_Shield	Power	Ground
45	SIM_DET	Digital I/O (SoC)	SIM detect
46	SIM_CLK	Digital I/O (SoC)	SIM clock
47	GND_Shield	Power	Ground
48	SIM_IO	Digital I/O (SoC)	SIM data
49	SIM_1V8	Power	SIM 1.8 V power supply output
50	GND_Shield	Power	Ground

Pin no	Pin name	Function	Description
51	Reserved		Do not connect/reserved for future use
52	GND_Shield	Power	Ground
53	MAGPIO2	Digital I/O (SoC)	1.8 V general purpose I/O
54	MAGPIO1	Digital I/O (SoC)	1.8 V general purpose I/O
55	MAGPIO0	Digital I/O (SoC)	1.8 V general purpose I/O
56	GND_Shield	Power	Ground
57	VIO	Power	MIPI RFFE control interface
58	SCLK	Digital I/O (SoC)	MIPI RFFE control interface
59	SDATA	Digital I/O (SoC)	MIPI RFFE control interface
60	GND_Shield	Power	Ground
61	ANT	RF	Single-ended 50 Ω LTE antenna pin
62	GND_Shield	Power	Ground
63	GND_Shield	Power	Ground
64	AUX	RF	Single-ended 50 Ω ANT loop-back pin
65	GND_Shield	Power	Ground
66	GND_Shield	Power	Ground
67	GPS	RF	Single-ended 50 Ω GPS input pin
68	GND_Shield	Power	Ground
69	GND_Shield	Power	Ground
70	Reserved		Do not connect/reserved for future use
71	Reserved		Do not connect/reserved for future use
72	GND_Shield	Power	Ground
73	Reserved		Do not connect/reserved for future use
74	GND_Shield	Power	Ground
75	GND_Shield	Power	Ground
76	GND_Shield	Power	Ground
77	GND_Shield	Power	Ground
78	GND_Shield	Power	Ground
79	GND_Shield	Power	Ground
80	GND_Shield	Power	Ground
81	GND_Shield	Power	Ground
82	GND_Shield	Power	Ground
83	P0.26	Digital I/O (SoC)	General purpose I/O
84	P0.27	Digital I/O (SoC)	General purpose I/O
85	GND_Shield	Power	Ground
86	P0.28	Digital I/O (SoC)	General purpose I/O
87	P0.29	Digital I/O (SoC)	General purpose I/O
88	P0.30	Digital I/O (SoC)	General purpose I/O
89	P0.31	Digital I/O (SoC)	General purpose I/O
90	GND_Shield	Power	Ground
91	COEX2	Digital I/O (SoC)	Coexistence interface
92	COEX1	Digital I/O (SoC)	Coexistence interface
93	COEX0	Digital I/O (SoC)	Coexistence interface
94	GND_Shield	Power	Ground
95	P0.00	Digital I/O (SoC)	General purpose I/O
96	P0.01	Digital I/O (SoC)	General purpose I/O
97	P0.02	Digital I/O (SoC)	General purpose I/O
98	GND_Shield	Power	Ground
99	P0.03	Digital I/O (SoC)	General purpose I/O
100	P0.04	Digital I/O (SoC)	General purpose I/O

Pin no	Pin name	Function	Description
101	ENABLE		Enable for the SiP internal regulator for the nRF91 SoC. **Note:** The nRF91 will not start until this pin is enabled.
102	VDD1	Power	Supply voltage
103	VSS	Power	Ground
104-127	Reserved		Do not connect/reserved for future use

Table 111: LGA pin assignments

10.2 Mechanical specifications

The mechanical specifications for the packages show the dimensions in millimeters.

10.2.1 16.00 x 10.50 mm package

Dimensions in millimeters for the nRF9160 LGA 16.00 x 10.50 mm package.

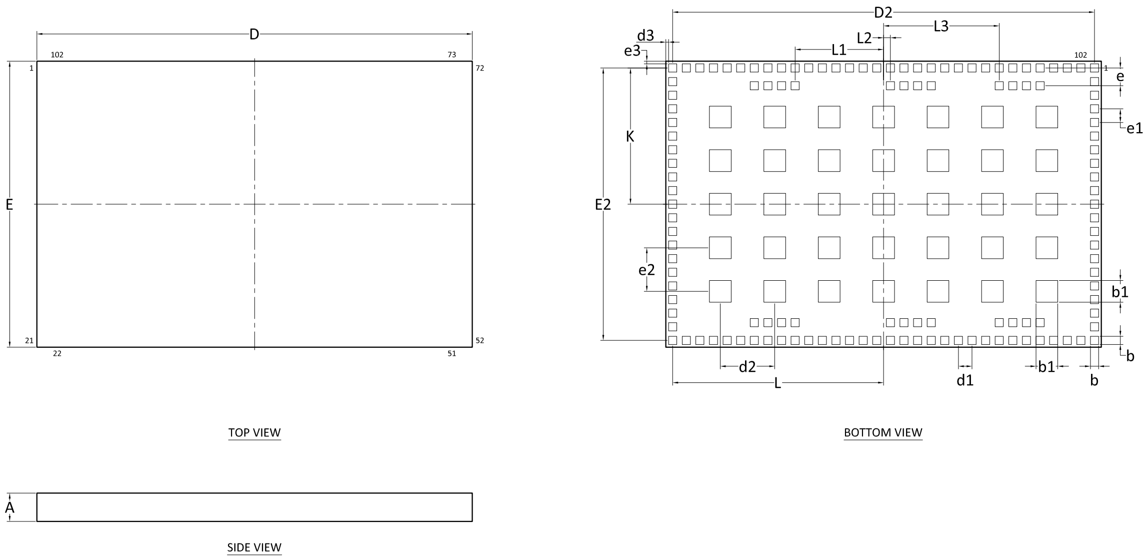

Figure 114: LGA 16.00 x 10.50 mm package

	A	b	b1	D	E	e	d1	e1	D2	E2	d2	e2	d3	e3	K	L	L1	L2	L3
Min.	0.98			15.90	10.40														
Nom.	1.04	0.30	0.80	16.00	10.50	0.65	0.50	0.50	15.50	10.00	2.00	1.60	0.10	0.10	5.00	7.75	3.25	0.25	4.25
Max.	1.10			16.10	10.60														

Table 112: LGA dimensions in millimeters

10.3 Reference circuitry

To ensure good RF performance when designing PCBs, it is highly recommended using the PCB layouts and component values provided by Nordic Semiconductor.

Documentation for the different package reference circuits, including Altium Designer files, PCB layout files, and PCB production files can be downloaded from the product page at www.nordicsemi.com.

In this section, there are reference circuits for SIxA to show the components and component values to support on-chip features in a design.

> **Note:** This is not a complete list of configurations, but all required circuitry is shown for further configurations.

10.3.1 Schematic SIxA LGA127

Circuit configuration for SIxA LGA127, showing the schematic and the Bill of Materials (BOM) table.

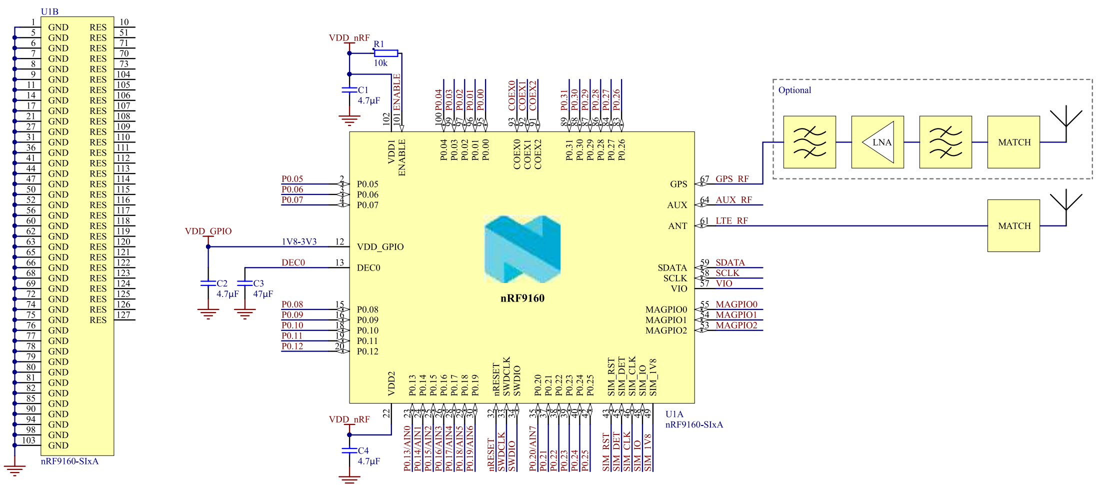

Figure 115: Schematic SIxA with antenna details

Designator	Value	Description	Footprint
C1, C4	4.7 µF	Capacitor, X5R, ±10%, 10 V	0603
C2	4.7 µF	Capacitor, X5R, ±10%, 6.3 V	0603
C3	47 µF	Capacitor, X5R, ±20%, 6.3 V	0805
R1	10 kΩ	Resistor, ±1%, 0.05 W	0201
U1	nRF9160-SIxA	Low power System-in-Package (SiP) with integrated LTE-M/NB-IoT modem and GPS	LGA

Table 113: BOM for SIxA LGA127

For PCB reference layouts, see the product page for the nRF9160 at www.nordicsemi.com.

11 Operating conditions

The operating conditions are the physical parameters that the chip can operate within.

Symbol	Parameter	Notes	Min.	Nom.	Max.	Units
VDD	Battery input voltage	Including voltage drop, ripple and spikes. RF 3GPP compliancy requires 3.3 V.	3.0	3.8	5.5	V
VDD_GPIO	GPIO input voltage		1.7		3.6	V
$GPIO_H$	GPIO high level voltage				VDD_GPIO	V
$MAGPIO_H$	MAGPIO high level voltage		1.7	1.8	1.9	V
VIO	VIO high level voltage		1.7	1.8	1.9	V
TA	Operating temperature		-40	25	85	°C

Table 114: Operating conditions

Note: There can be excessive leakage at VDD and/or VDD_GPIO if any of these supply voltages is outside its range given in the table above.

Note: It is not recommended to use high voltage, high drive GPIO outputs ($V_{OH,HDH}$ and $V_{OH,HDL}$) with high frequency, high capacitance loads unless needed, as this may increase noise level and affect radio receiver performance. High drive/high load should especially be avoided on GPIO pins close to the radio front end.

11.1 VDD_GPIO considerations

VDD_GPIO is the supply to the general purpose I/O.

The following restrictions should be taken into considerations:

- VDD_GPIO should be applied after VDD has been supplied
- VDD_GPIO should be removed before removing VDD
- If VDD is supplied and VDD_GPIO is grounded, an extra current consumption can be generated on VDD
- If ENABLE is low, VDD_GPIO should also be low

12 Absolute maximum ratings

Maximum ratings are the extreme limits to which the chip can be exposed for a limited amount of time without permanently damaging it. Exposure to absolute maximum ratings for prolonged periods of time may affect the reliability of the device.

	Note	Min.	Max.	Unit
Supply voltages				
VDD		-0.3	5.5	V
VDD_GPIO		-0.3	3.9	V
SIM_1V8		1.65	1.95	V
VSS			0	V
I/O pin voltage				
$V_{I/O}$, VDD_GPIO ≤ 3.6 V		-0.3	VDD_GPIO + 0.3	V
$V_{I/O}$, VDD_GPIO > 3.6 V		-0.3	3.9	V
Radio				
RF input level			10	dBm
Environmental (LGA package)				
Storage temperature		-40	125	°C
MSL	Moisture Sensitivity Level		3	
ESD HBM	Human Body Model		1.5	kV
ESD HBM Class	Human Body Model Class		1C	
ESD CDM	Charged Device Model		500	V
Flash memory				
Endurance		10 000		Write/erase cycles
Retention		10 years at 85°C		

Table 115: Absolute maximum ratings

13 Ordering information

This chapter contains information on IC marking, ordering codes, and container sizes.

13.1 IC marking

The nRF9160 IC package is marked like described below.

N	9	1	6	0	
<P	P>	<V	V>	<H>	<P>
<Y	Y>	<W	W>	<L	L>

Figure 116: Package marking

13.2 Box labels

Here are the box labels used for the nRF9160.

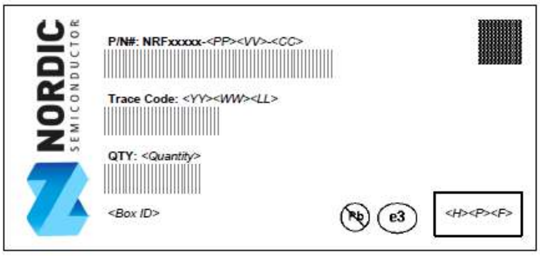

Figure 117: Inner box label

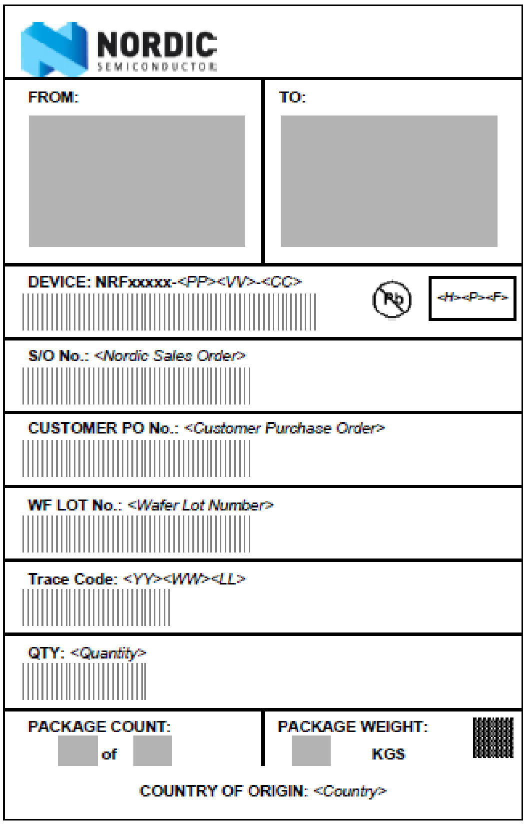

Figure 118: Outer box label

13.3 Order code

Here are the nRF9160 order codes and definitions.

n	R	F	9	1	6	0	-	<P	P>	<V	V>	-	<C	C>

Figure 119: Order code

Abbreviation	Definition and implemented codes
N91/nRF91	nRF91 Series product
60	Part code
<PP>	Package variant code
<VV>	Function variant code
<H><P><F>	Build code H - Hardware version code P - Production configuration code (production site, etc.) F - Firmware version code (only visible on shipping container label)
<YY><WW><LL>	Tracking code YY - Year code WW - Assembly week number LL - Wafer lot code
<CC>	Container code

Table 116: Abbreviations

13.4 Code ranges and values

Defined here are the nRF9160 code ranges and values.

<PP>	Package	Size (mm)	Pin/Ball count	Pitch (mm)
SI	LGA	16.00 x 10.50	127	0.50

Table 117: Package variant codes

<VV>	LTE-M/NB-IoT/GPS	Flash (kB)	RAM (kB)
CA	LTE-M/NB-IoT/GPS	1024	256
AA	LTE-M only		
BA	NB-IoT only		

Table 118: Function variant codes

<H>	Description
[A . . Z]	Hardware version/revision identifier (incremental)

Table 119: Hardware version codes

<P>	Description
[0 . . 9]	Production device identifier (incremental)
[A . . Z]	Engineering device identifier (incremental)

Table 120: Production configuration codes

<F>	Description
[A . . N, P . . Z]	Version of preprogrammed firmware
[0]	Delivered without preprogrammed firmware

Table 121: Production version codes

<YY>	Description
[15 . . 99]	Production year: 2015 to 2099

Table 122: Year codes

<WW>	Description
[1 . . 52]	Week of production

Table 123: Week codes

<LL>	Description
[AA . . ZZ]	Wafer production lot identifier

Table 124: Lot codes

<CC>	Description
R7	7" Reel
R	13" Reel
T	Tray

Table 125: Container codes

13.5 Product options

Defined here are the nRF9160 product options.

Order code	Minimum ordering quantity (MOQ)	Comment
nRF9160-SICA-R	2500	LTE-M/NB-IoT/GPS product
nRF9160-SICA-R7	100	LTE-M/NB-IoT/GPS product
nRF9160-SIAA-R	2500	LTE-M only product
nRF9160-SIAA-R7	100	LTE-M only product
nRF9160-SIBA-R	2500	NB-IoT only product
nRF9160-SIBA-R7	100	NB-IoT only product

Table 126: nRF9160 order codes

Order code	Description
nRF9160-DK	nRF9160 Development Kit
nRF6943	Nordic Thingy:91

Table 127: Development tools order code

14 Regulatory information

The nRF9160 undergoes a number of regulatory certifications, ensuring both regional compliancies and compatibility with the LTE 3GPP specification.

For information about certified bands, and status for the ongoing certifications, see nRF9160 Certifications.

15 Legal notices

By using this documentation you agree to our terms and conditions of use. Nordic Semiconductor may change these terms and conditions at any time without notice.

Liability disclaimer

Nordic Semiconductor ASA reserves the right to make changes without further notice to the product to improve reliability, function, or design. Nordic Semiconductor ASA does not assume any liability arising out of the application or use of any product or circuits described herein.

Nordic Semiconductor ASA does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. If there are any discrepancies, ambiguities or conflicts in Nordic Semiconductor's documentation, the Product Specification prevails.

Nordic Semiconductor ASA reserves the right to make corrections, enhancements, and other changes to this document without notice.

Life support applications

Nordic Semiconductor products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury.

Nordic Semiconductor ASA customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Nordic Semiconductor ASA for any damages resulting from such improper use or sale.

RoHS and REACH statement

Refer to www.nordicsemi.com for complete hazardous substance reports, material composition reports, and latest version of Nordic's RoHS and REACH statements.

Trademarks

All trademarks, service marks, trade names, product names, and logos appearing in this documentation are the property of their respective owners.

Copyright notice

COMPANY WITH
QUALITY SYSTEM
CERTIFIED BY DNV GL
= ISO 9001 =